Jül-4377

Institute for Advanced Simulation (IAS)
Jülich Supercomputing Centre (JSC)

Spectral Deferred Corrections für das Slow-Wave-Fast-Wave-Problem

Marina Weingartz

Member of the Helmholtz Association

JÜLICH
FORSCHUNGSZENTRUM

Spectral Deferred Corrections für das Slow-Wave-Fast-Wave-Problem

Marina Weingartz

Berichte des Forschungszentrums Jülich; 4377
ISSN 0944-2952
Institute for Advanced Simulation (IAS)
Jülich Supercomputing Centre (JSC)
Jül-4377

Vollständig frei verfügbar im Internet auf dem Jülicher Open Access Server (JUWEL)
unter http://www.fz-juelich.de/zb/juwel

Zu beziehen durch:
Forschungszentrum Jülich GmbH
Zentralbibliothek, Verlag
52425 Jülich
Tel.: +49 2461 61-5220
Fax: +49 2461 61-6103
E-Mail: zb-publikation@fz-juelich.de
www.fz-juelich.de/zb

Zusammenfassung

Im Bereich der Meteorologie treten verschiedene atmosphärische und ozeanische Wellen mit sehr unterschiedlichen Dynamiken auf. Um numerische Verfahren bzgl. dieser Problemstellung zu analysieren, ist die Slow-Wave-Fast-Wave-Gleichung eine geeignete Testgleichung. Es existieren bereits Ansätze basierend auf einfachen Verfahren, wie dem Trapez-Leapfrog-Verfahren und den Adams-Verfahren. Diese weisen jedoch Probleme hinsichtlich der Stabilität und der Genauigkeit auf. In dieser Arbeit wurde ein flexibles, numerisches Lösungsverfahren namens Spectral Deferred Corrections (SDC) vorgestellt und untersucht. Dabei wurden die schnelle und die langsame Dynamik mit verschiedenen Ansätzen behandelt, womit stabile und genaue Ergebnisse der Slow-Wave-Fast-Wave-Gleichung möglich sind. Dieses Verfahren hoher Ordnung nutzt ein einfaches numerisches Basis-Verfahren, z. B. ein Euler-Verfahren, zur Bestimmung einer vorläufigen Approximation und verbessert diese Näherungslösung iterativ durch das Lösen einer Folge von Korrekturgleichungen. Im Rahmen dieser Arbeit wurden mehrere Varianten von SDC auf Konvergenz, Stabilität und Genauigkeit hin für die allgemeine Testgleichung untersucht. Im Anschluss daran wurde SDC für die Slow-Wave-Fast-Wave-Gleichung analysiert. Abschließend beinhaltet diese Arbeit eine Kosten-Nutzen-Analyse des SDC-Verfahrens im Vergleich zum Trapez-Leapfrog- und zum kombinierten Adams-Verfahren angewandt auf die SWFW-Gleichung. Dabei k”pnnen einige wichtige Erkenntnisse gewonnen werden, die in vielerlei Hinsicht weiterverwendet und -entwickelt werden können.

Abstract

In the wide range of meteorology, there exist a lot of different atmospheric and oceanic waves which exhibit varying dynamics. In order to analyse numerical methods with respect to such a problem, the so-called Slow-Wave-Fast-Wave-Equation is an appropriate test equation. There already exist some approaches based on simple methods like IMEX leapfrog-trapezoidal approximations or the IMEX adams approximations. Using this methods, causes problems in stability and accuracy. In this publication a flexible, numerical method named Spectral Deferred Corrections (SDC) was presented and analysed. Within this work the fast and the slow dynamic was dealt with different approaches. This lead to stable and accurate results for the Slow-Wave-Fast-Wave-Equation. The high order method, SDC, is based on a simple numerical base-method, e. g. an Euler-method, to calculate a provisional solution and improves this approximation iteratively by solving a sequence of correction equations. Within the scope of this work various implementation variants of SDC were examined with respect to convergence, stability and accuracy for the common test equation. After this, there is an analysis of SDC applayed to the Slow-Wave-Fast-Wave-Equation. Finally, this work includes an analysis of the cost of the SDC-method compared to the leapfrog-trapezoidal- and the combined adams-method applayed to the SWFW-Equation. This leads to some important knowledge and insights which can be further used and developed in many ways.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1. Einleitung und Motivation

Die vorliegende Arbeit beschäftigt sich mit der Lösung gewöhnlicher Differentialgleichungen, die in Form von Anfangswertproblemen gegeben sind, durch ausgewählte numerische Lösungsverfahren.
Differentialgleichungen sind Gleichungen, in denen neben der gesuchten Funktion auch ihre Ableitung/en vorkommen. Sie kommen meist bei der Modellierung naturwissenschaftlicher Phänomene, Vorgänge etc. zum Einsatz. Oft wird durch solch eine Gleichung das Änderungsverhalten physikalischer Größen beschrieben. Es gibt viele verschiedene Typen von Differentialgleichungen, die in diversen mathematischen Modellierungsgebieten genutzt werden können.

1.1. Motivation

In dieser Arbeit werden hauptsächlich solche Differentialgleichungen in den Vordergrund gestellt, die sich aus einem Gleichungsanteil von schneller Dynamik und aus einem Gleichungsanteil von langsamer Dynamik zusammensetzen. Dieser Typus von Differentialgleichung, der im Verlauf der Arbeit näher analysiert wird, wird in diversen Modellierungsgebieten sinnvoll angewendet. Ein wesentliches Einsatzgebiet solcher Differentialgleichungen ist die Meteorologie. Die Atmosphäre und der Ozean erzeugen verschiedenartige Wellen, die sich mit unterschiedlichen Geschwindigkeiten ausbreiten. Atmosphärische Modelle umfassen sowohl niedrigfrequente Wellen wie die Rossby-Wellen[1], Schwerewellen[2] und einfache Advektion[3] als auch hochfrequente Wellen wie Lamb-Wellen[4] und akustische Wellen.
Anwendungen, in denen solche Beschreibungen des komplexen atmosphärischen Systems vorkommen, sind z. B. das fifth-generation Pennsylvania State University-National-Center for Atmospheric Research Mesoscale Model, das Advanced Research Weather Research and Forecast (WRF)-NCAR Model, das Advanced Regional Prediction System oder das lokale Modell des Deutschen Wetterdienstes [6].
Auch die Shallow-Water-Gleichungen, die unter anderem zur Modellierung von küstennahen Ozeanstellen oder Seen, Flusssimulationen in Kanälen oder Flüssen etc. genutzt werden, umfassen Wellen von stark variierender Frequenz [7, 8].
Ferner werden Differentialgleichungen mit Gleichungsanteilen von unterschiedlicher Dynamik, die spezielle Lösungstechniken erfordern, in Modellrechnungen zur Klima-Simulation eingesetzt [9].

[1] So werden großräumige Wellenbewegungen in der Erdatmosphäre und windgetriebene Ozeanzirkulationen bezeichnet [1].

[2] Die Wellen, die sich an der Oberfläche von Wasser in der Umgebung des Wellenursprungs ausbreiten, werden als Schwerewellen bezeichnet. Die wellenartige Ausbreitung hängt zusammen mit der Trägheit und der Schwere des Wassers [2]. Zusätzlich gibt es Schwerewellen, die durch die Schichtung der Atmosphäre entstehen. Vor allem diese sind für Meteorologen von Bedeutung [3].

[3] Die horizontale Bewegung der Luft bzw. ein Transportprozess, der nur in eine Richtung abläuft, wird oft Advektion genannt [4].

[4] Lamb-Wellen werden oft zur Erkennung von Strukturschäden durch Schwingungen benutzt. Dabei handelt es sich um Schwingungen einer Platte. Zudem können Lamb-Wellen zur Analyse von Unterwasserexplosionen und einigen anderen Dingen verwendet werden [5].

Neben den gerade schon zitierten Arbeiten von Gassmann und Herzog [6], Bourchtein [7], Whitaker [8] und Wicker [9] haben auch Jebens, Knoth und Weiner [10], Baldauf [11], Kar [12] und erneut Gassmann [13] Ausarbeitungen über die Analyse von Lösungsmethoden für Differentialgleichungen mit Gleichungsanteilen unterschiedlicher Dynamik, die im Bereich der Meteorologie Verwendung finden, veröffentlicht.

Zur sinnvollen Lösung solcher Differentialgleichungen, die sowohl Terme von schneller Dynamik als auch Terme von langsamer Dynamik umfassen, sind spezielle Lösungstechniken erforderlich. Da Wellen mit hoher Frequenz eine wesentlich feinere Aufteilung des Zeitintervalls benötigen, muss zu einer stabilen und genauen Integration von sich schnell ausbreitenden Wellen ein sehr kleiner Zeitschritt verwendet werden [14]. Generell wird der größte Zeitschritt, für den das Verfahren stabil ist, bei Gleichungen, die den atmosphärischen Fluss approximativ berechnen, durch die Geschwindigkeit der sich schnell ausbreitenden Welle bestimmt [15]. In der Realität auftretende Wellen, die sich sehr schnell ausbreiten, sind Schallwellen. Schallwellen spielen bei Strömungen sowohl in der Atmosphäre als auch im Ozean keine große Rolle und sind somit in der Genauigkeit der Modellierung vernachlässigbar [15].
Jedoch erreichen Jetstreams[5], die in der numerischen Wettervorhersage eine große Bedeutung haben, bis zu einem Drittel der Schallgeschwindigkeit [16] und erschweren durch ihre hohe Ausbreitungsgeschwindigkeit die Lösung der zur Modellierung benötigten Differentialgleichungen.

Einfache numerische Verfahren haben enorme Probleme solche Differentialgleichungen zu lösen, die sich aus einer schnell und einer langsam ausbreitenden Welle zusammensetzen. Die auftretenden Probleme werden im Folgenden an der 'Testgleichung' für die gerade erläuterten Eigenschaften einer Differentialgleichung, der Slow-Wave-Fast-Wave-Gleichung, dargestellt.

1.2. Slow-Wave-Fast-Wave-Problem

Eine spezielle Gleichung, mit der das Verhalten einer sich langsam und einer sich schnell ausbreitenden Welle simuliert werden kann, nennt sich Slow-Wave-Fast-Wave-Gleichung[6] und wird in diesem Kapitel genauer beschrieben. Die SWFW-Gleichung kann als Testgleichung für Differentialgleichungen verstanden werden, die Wellen von stark unterschiedlichen Frequenzen modellieren. Diese Testgleichung kann als Summe von zwei komplexen Zahlen geschrieben werden. Aus der Struktur dieser einfachen Testgleichung, die in der Stabilitäts- und Genauigkeitsanalyse verwendet wird, lassen sich viele komplexere und speziellere Differentialgleichungen herleiten. Die Nutzung einer auf das konkrete Problem hin ausgewählten Testgleichung dient in vielen Fällen zur Klassifikation verschiedener Lösungsverfahren oder zur Beurteilung von deren Qualität.

Die sogenannte SWFW-Gleichung ist eine Variante der Schwingungsgleichung. Sie stellt eine

[5] Jetstreams sind sehr starke Windbänder, die vor allem in Regionen mit starken Temperaturunterschieden auftreten [1].

[6] Im Folgenden wird für den Begriff 'Slow-Wave-Fast-Wave' die Abkürzung 'SWFW' verwendet.

Differentialgleichung dar, die sich als Summe einer sich schnell ausbreitenden Welle und einer sich langsam ausbreitenden Welle schreiben lässt. Die schnelle Welle der SWFW-Gleichung besitzt nur eine relativ kleine Frequenz und braucht deshalb zur Simulation nicht allzu kleine Zeitschritte [15].

Bei den im Rahmen dieser Arbeit durchgeführten Stabilitäts-, Genauigkeits- und Konvergenzuntersuchungen wird die SWFW-Gleichung als Testgleichung für die semi-implizite Implementierungsweise genutzt. Die allgemein als Testgleichung bezeichnete Differentialgleichung erster Ordnung ist durch

$$u'(t) = \lambda \cdot u(t) \tag{1}$$

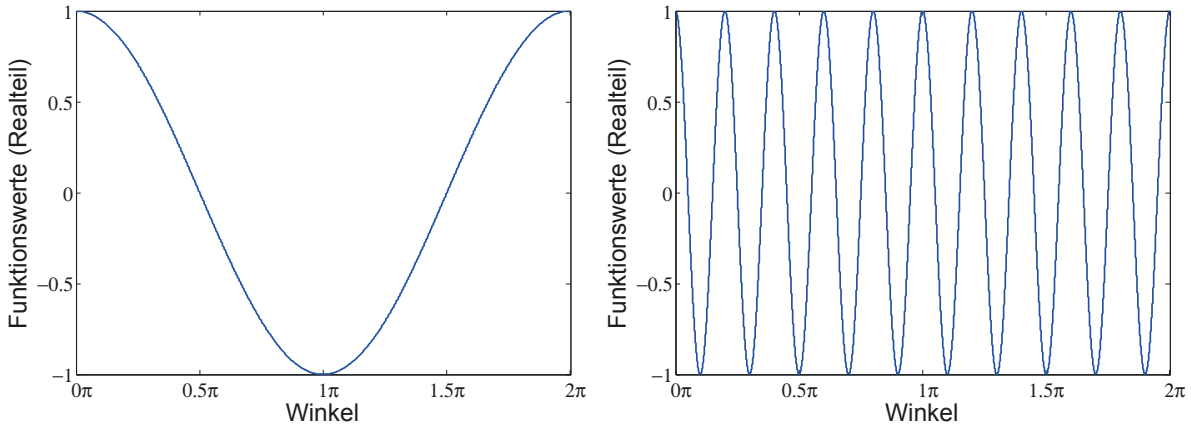
mit exakter Lösung $u(t) = e^{\lambda t}$ zum Anfangswert $u(0) = 1$ gegeben, wobei $\lambda \in \mathbb{C}$ und $t \in \mathbb{R}^{\geq 0}$.

Die SWFW-Gleichung baut auf dieser elementaren Testgleichung auf. Der Unterschied besteht darin, dass man hier anstatt $\lambda = -1$, was häufig bei der elementaren Stabilitätsanalyse verwendet wird, oder $\lambda = a + \imath b$ mit $a, b \in \mathbb{R}$, was im Allgemeinen zur Stabilitätsanalyse numerischer Verfahren genutzt wird, $\lambda = \imath w_H + \imath w_L$ verwendet.[7] Damit erhält man die Gleichung

$$u'(t) = \lambda \cdot u(t) = (\imath w_H + \imath w_L) \cdot u(t) = \imath w_H \cdot u(t) + \imath w_L \cdot u(t) \tag{2}$$

mit exakter Lösung $u(t) = e^{\imath w_H \cdot t} \cdot e^{\imath w_L \cdot t}$ zum Anfangswert $u(0) = 1$.

Zur besseren bildlichen Vorstellung der SWFW-Gleichung ist in Abbildung 1 der Anteil der langsamen Welle und der Anteil der schnellen Welle der Differentialgleichung getrennt dargestellt.



(a) Realteil der langsamen Welle der Differentialgleichung: $e^{\imath w_L \cdot t}$ für $w_L = 1$

(b) Realteil der schnellen Welle der Differentialgleichung: $e^{\imath w_H \cdot t}$ für $w_H = 10$

Abbildung 1: Aufspaltung der SWFW-Gleichung

[7] Wobei $\imath$ die imaginäre Einheit darstellt.

Es ist gut zu erkennen, dass die Zeitskalen der beiden Lösungsanteile sehr unterschiedlich sind. Man müsste also zur genauen Berechnung der schnellen Welle in Teilgrafik (b) sehr kleine Zeitschritte wählen. Diese Eigenschaft ist ein entscheidendes Merkmal für ein steifes System von Differentialgleichungen. Generell spricht man von einem steifen Differentialgleichungssystem, wenn es in den Lösungsanteilen stark unterschiedliche Zeitskalen gibt [17].[8] Je größer der Unterschied zwischen dem schnellen und dem langsamen Gleichungsanteil ist, desto größer werden die Probleme bei der numerischen Lösung der SWFW-Gleichung.
Über den Zeitraum von 0 bis 2π entsprechen die Koeffizienten $w_H\Delta t$ und $w_L\Delta t$ der Anzahl der Perioden der Schwingung, wobei Δt die Größe des Zeitschritts innerhalb des numerischen Verfahrens angibt.[9] Zudem sind die Schwingungswellen symmetrisch. Das bedeutet, dass die Wellen für $w_H = -10$ und $w_L = -1$ identisch mit den Schwingungswellen in Abbildung 1 sind.[10]
Abschließend zeigt Abbildung 2 die exakte Lösung von Gleichung (2) zu $w_L = 1$ und $w_H = 10$ zum Anfangswert $u(0) = 1$. Zu sehen ist somit die Kombination aus Teil (a) und Teil (b) von Abbildung 1. Auch hier wird – äquivalent zu den zwei voranstehenden Abbildungen – nur der Realteil der Lösung abgebildet. Zur Darstellung des Realteils des exakten Ergebnisses der SWFW-Gleichung zu $w_L = 1$ und $w_H = 10$ wird $cos((1+10)t) = cos(11t)$ berechnet. Daraus resultiert eine Schwingung mit elf Perioden über den Zeitraum von 0 bis 2π. Das ist genau eine Periode mehr als bei der separierten Darstellung der Schwingung der schnellen Welle.

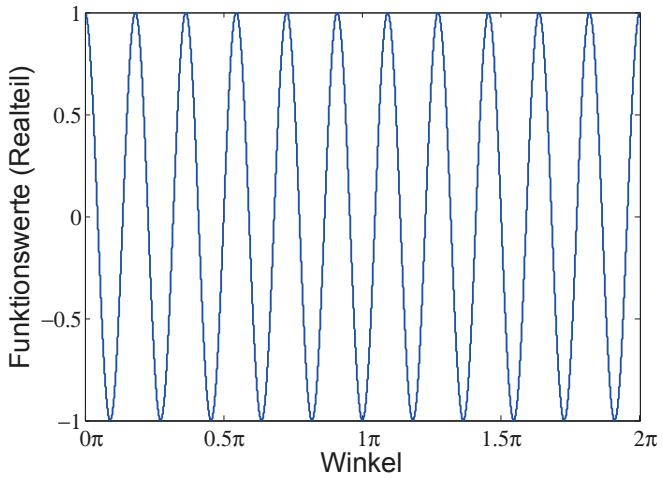


Abbildung 2: Realteil des exaktes Ergebnisses für die SWFW-Gleichung als Kombination der Lösungen für den schnellen und den langsamen Gleichungsanteil

[8] Im Allgemeinen versteht man unter einem steifen Problem meist Probleme mit sehr kleinen reellen Eigenwerten: $Re(\lambda) \ll 0$ [18]. Das SWFW-Problem ist da zwar anders, aber aufgrund der vorkommenden unterschiedlichen Geschwindigkeiten in der Dynamik bildet es dennoch ein steifes Problem.

[9] In dieser Auswertung ist $\Delta t = 1$ gewählt. Somit entsprechen $w_H\Delta t, w_L\Delta t$ den Ausdrücken w_H, w_L.

[10] Da hier nur der Realteil der Lösung abgebildet ist, entspricht die dargestellte Schwingung der langsamen Welle $cos(1t)$ und die dargestellte Schwingung der schnellen Welle $cos(10t)$. Die Imaginärteile weisen ein ebenso unterschiedliches Verhalten bezüglich der Dynamik auf.

Es stellt sich die Frage, ob diese Gleichung mit einem billigen und einfachen expliziten numerischen Verfahren von geringem Aufwand gelöst werden kann. Die Antwort liegt jedoch auf der Hand. Da mit der SWFW-Gleichung eine steife Differentialgleichung vorliegen kann,[11] würde in diesem Fall ein sehr kleiner Zeitschritt in der numerischen Auswertung benötigt werden, um Stabilität sicherstellen zu können [14].

Explizite Löser sind nicht A-stabil und aufgrund der Stabilitätsprobleme steifer Gleichungen nicht sinnvoll auf diese anwendbar. Anstelle dessen müssen daher implizite numerische Lösungsverfahren benutzt werden. Diese sind durch die enthaltenen impliziten Funktionsauswertungen wesentlich aufwendiger, bilden allerdings eine brauchbare Alternative hinsichtlich steifer Differentialgleichungen.[12] Steife Terme haben nur bei der Anwendung eines impliziten Lösungsverfahrens und realisierbar kleinen Schrittweiten eine stabile Lösung [19].[13]

Es liegt nun das SWFW-Problem als Ausgangspunkt vor. Dieses soll auf eine sinnvolle Art und Weise gelöst werden. Dazu wurden diverse Ansätze entwickelt.

Durran und Blossey [15] haben in ihrer Veröffentlichung vom April 2012 zunächst einen einfachen Ansatz gewählt, indem sie das Trapez-Leapfrog-Verfahren und das Zweischritt-Adams-Bashforth-Verfahren auf Stabilität bei der approximativen Lösung der SWFW-Gleichung hin untersucht haben. Dies liefert den Ausgangspunkt für diese Arbeit. Nach [15] ist das Trapez-Leapfrog IMEX[14]-Verfahren das Verfahren, welches in der atmosphärischen Wissenschaft verbreitet genutzt wird.

Generell können u. a. atmosphärische und ozeanische Welle in der Form

$$u'(t) = f(u) + Lu \tag{3}$$

geschrieben werden. u ist hierbei die Zustandsvariable. L ist die Diskretisierung eines linearen, räumlichen Operators, welcher schnelle Prozesse – wie die Schallgeschwindigkeit – modelliert. Alle anderen Gleichungsterme – wie z. B. Advektion – werden durch $f(u)$ ausgedrückt [15].
Sei nun q^n eine Näherungslösung für u zum Zeitpunkt $n\Delta t$, dann lassen sich diverse einfache semi-implizite numerische Verfahren zur Lösung von Gleichung (3) herleiten. Diese kombinieren ein explizites und ein implizites Lösungsverfahren, um die langsamen und schnellen Gleichungsanteile aus Gleichung (3) separat behandeln zu können.

[11] Dies ist der Fall, wenn die Koeffizienten innerhalb der Gleichung einen nennenswerten Größenunterschied aufweisen.
[12] In der Regel müssen hier nichtlineare Gleichungssysteme gelöst werden.
[13] Es gibt zwar explizite Methoden, die z. B. zur Lösung des Problems unterschiedlich große Schrittweiten für w_H und w_L verwenden. Diese Vorgehensweise ruiniert allerdings die Ordnung solcher Verfahren [9].
[14] Semi-implizite Verfahren werden oft auch implizit-explizite Verfahren (IMEX) genannt.

1.2.1. Trapez-Leapfrog- und Adams-Bashforth-Verfahren

Konstruiert wird nun einerseits der Trapez-Leapfrog-Algorithmus, indem das implizite Trapez-Verfahren mit dem expliziten Leapfrog-Verfahren kombiniert wird. Die Kombination aus beiden Verfahren kann als Prädiktor-Korrektor-Methode angesehen werden [20, 21]. Man beginnt mit einem Prädiktor-Schritt, der durch das einfache explizite Leapfrog-Verfahren realisiert wird. Dabei wird ein q^{n+1} mit der Rechenvorschrift

$$q^{n+1} = q^{n-1} + 2\Delta t \cdot f(q^n) \tag{4}$$

ermittelt, wobei $f(q^n)$ hier erneut für die nicht schnellen Gleichungsanteile steht [15, 20]. Formt man die Gleichung des Leapfrog-Verfahrens ein wenig um, so kann diese durch

$$\frac{q^{n+1} - q^{n-1}}{2\Delta t} = f(q^n) \tag{5}$$

ausgedrückt werden.

Neben dem Prädiktor wird ein Korrektor verwendet, der mit Hilfe des Trapez-Verfahrens realisiert wird. Das einfache implizite Trapez-Verfahren über den Zeitraum von Δt lautet

$$q^{n+1} = q^n + \frac{\Delta t}{2}(Lq^n + Lq^{n+1}). \tag{6}$$

Dieses einfache Trapez-Verfahren ist jedoch nicht in der Lage eine Dämpfung der Wellen aus der gegeben Differentialgleichung zu berücksichtigen [15]. Aus diesem Grund wird das Trapez-Verfahren auf ein gewichtetes Trapez-Verfahren erweitert. Dabei wird die Variable θ mit $0 \leq \theta \leq 1$ eingeführt, welche die Stärke der Dämpfung der numerischen Lösung steuert. Durch die Integration dieser Variable in Gleichung (6) erhält man

$$q^{n+1} = q^n + \Delta t(\theta Lq^n + (1-\theta)Lq^{n+1}), \tag{7}$$

das so genannte gewichtete Trapez-Verfahren.[15]
Bei der Wahl der Variable θ ist hier das Verhältnis von Dämpfung und Genauigkeit abzuwägen. Mit $\theta \approx \frac{1}{2}$ erfolgt nur eine schwache Dämpfung, aber dafür eine hohe Genauigkeit [15]. Laut Durran wird Dämpfung oft dazu benötigt, störende und gefälschte Oszillationen zu vermeiden. Bei der Lösung des SWFW-Problems ist es manchmal sinnvoll die höchsten vorkommenden Frequenzen etwas zu dämpfen, um Stabilität bei der Lösung zu gewährleisten und genauere Lösungen zu erhalten [15].
In den folgenden Auswertungen der Arbeit wird immer mit $\theta = \frac{1}{2}$ gerechnet und das gewichtete Trapez-Verfahren ist dann durch die Gleichung (6) gegeben. Um Kompatibilität zwischen dem Leapfrog-Verfahren, welches über einem Zeitraum von $2\Delta t$ arbeitet, und dem Trapez-Verfahren herzustellen, wird nun auch das Trapez-Verfahren über einen Zeitraum von $2\Delta t$ betrachtet. Somit ergibt sich die Iterationsvorschrift

$$q^{n+1} = q^{n-1} + \frac{2\Delta t}{2}(Lq^{n+1} + Lq^{n-1}), \tag{8}$$

[15] Wählt man $\theta = 1$, erhält man mit Gleichung (7) das explizite Euler-Verfahren. Bei der Verwendung von $\theta = 0$ ergibt sich das implizite Euler-Verfahren. Das eigentliche Trapez-Verfahren entsteht durch die Wahl von $\theta = \frac{1}{2}$.

die zur Gleichung

$$\frac{q^{n+1}-q^{n-1}}{2\Delta t} = \frac{1}{2}(Lq^{n+1}+Lq^{n-1}) \tag{9}$$

umgeformt werden kann. Dabei entspricht die linke Seite von Gleichung (9) der linken Seite von Gleichung (5). Durch eine Kombination aus beiden gerade genannten Gleichungen erhält man

$$\frac{q^{n+1}-q^{n-1}}{2\Delta t} = f(q^n)+\frac{1}{2}(Lq^{n+1}+Lq^{n-1}), \tag{10}$$

das Trapez-Leapfrog-Verfahren [15, 20]. Ein Vorteil der Verknüpfung dieser beiden Verfahren ist die Erweiterung des auf einer Schrittweite arbeitenden Trapez-Verfahrens auf ein zweistufiges Trapez-Leapfrog-Verfahren.

Die SWFW-Gleichung – die Testgleichung mit $\lambda = \imath w_H + \imath w_L$ – liefert konkrete Werte zur Anwendung des Trapez-Leapfrog-Verfahrens. Der schnelle Wellenanteil, der durch $\imath w_H$ verkörpert wird, soll nun implizit behandelt und der langsame Wellenanteil, der durch $\imath w_L$ gegeben ist, soll explizit gelöst werden. Demzufolge erhält man für die allgemeine Gleichung (10) im konkreten Anwendungsfall für die SWFW-Gleichung

$$\frac{q^{n+1}-q^{n-1}}{2\Delta t} = \imath w_L \cdot q^n + \frac{1}{2}\imath w_H(q^{n+1}+q^{n-1}) \tag{11}$$

als finale Iterationsvorschrift für das Trapez-Leapfrog-Verfahren.

Ein anderer Lösungsansatz bietet das klassische dreistufige explizite Adams-Bashforth-Verfahren in Kombination mit einem zweistufigen impliziten Adams-Verfahren. Dieses kombinierte, semi-implizite Lösungsverfahren kann als Dreischritt-Verfahren zweiter Ordnung konstruiert werden [15]. Das klassische, dreistufige Adams-Bashforth-Verfahren ist durch die Iterationsvorschrift

$$q^{n+1} = q^n + \Delta t(\frac{23}{12}f(q^n) - \frac{4}{3}f(q^{n-1}) + \frac{5}{12}f(q^{n-2})) \tag{12}$$

gegeben [20]. Dabei beinhaltet f, wie auch schon beim gewichteten Trapez-Leapfrog-Verfahren, alle nicht schnellen Gleichungsanteile.

Die Gleichung

$$q^{n+1} = q^n + \frac{1}{2}\Delta t((1+c)Lq^{n+1} + (1-2c)Lq^n + cq^{n-1}), \tag{13}$$

mit $c \in [0;1]$, stellt die allgemeine Rechenvorschrift für die einparametrische Familie von impliziten Adams-Methoden dar. Wählt man in Gleichung (13) für $c = \frac{3}{2}$, so erhält man ein zweistufiges implizites Adams-Verfahren [15]. Konkret ergibt sich die Gleichung

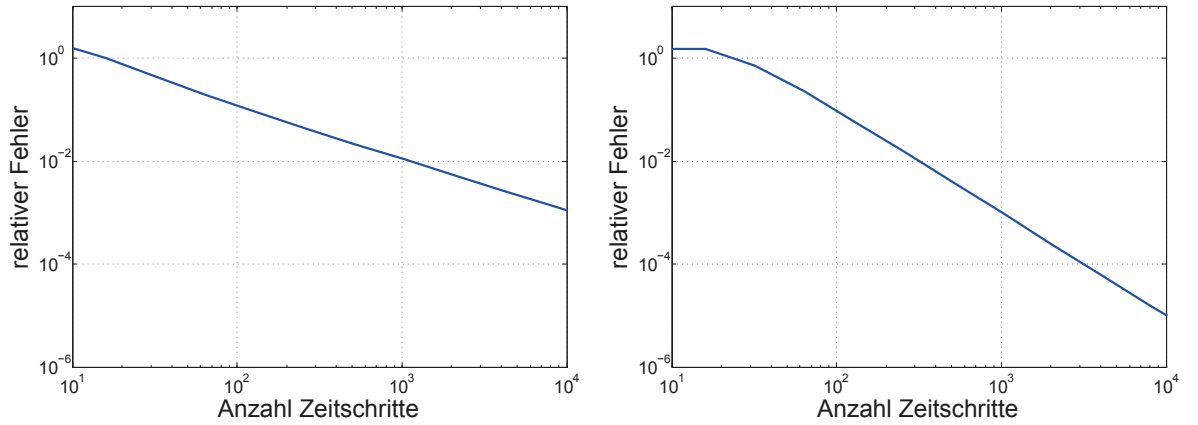
$$q^{n+1} = q^n + \Delta t(\frac{5}{4}Lq^{n+1} - Lq^n + \frac{3}{4}Lq^{n-1}) \tag{14}$$

als Iterationsvorschrift für die Gleichung (13) mit $c = \frac{3}{2}$ [15]. Auch hier ist L die Diskretisierung eines linearen, räumlichen Operators, welcher schnelle Prozesse beinhaltet.
Diese beiden Verfahren lassen sich nun zu einem kombinierten Verfahren zusammensetzen, welches aus einem impliziten und einem expliziten Löser besteht. Wenn man die Testgleichung mit $\lambda = (\imath w_H + \imath w_L)$ betrachtet, so ist w_H, der Gleichungsanteil mit schneller Dynamik, implizit zu lösen und w_L, der Gleichungsanteil von langsamer Dynamik, explizit lösbar. Basierend auf diesem Hintergrund können die Summanden von λ in eine kombinierte Iterationsvorschrift eingesetzt werden und es ergibt sich

$$\frac{q^{n+1} - q^n}{\Delta t} = \imath w_H(\frac{5}{4}q^{n+1} - q^n + \frac{3}{4}q^{n-1}) + \imath w_L(\frac{23}{12}q^n - \frac{4}{3}q^{n-1} + \frac{5}{12}q^{n-2}) \tag{15}$$

als finale Iterationsvorschrift [15].
Mit Hilfe dieser beiden Iterationsvorschriften kann vorab eine kurze Stabilitätsanalyse des Trapez-Leapfrog-Verfahrens und des Adams-Bashforth-Verfahrens hinsichtlich der SWFW-Gleichung durchgeführt werden.

Betrachtet man nun den relativen Fehler aufgetragen zur Anzahl der Zeitschritte, in die das Intervall $[0;1]$ unterteilt wird,[16] so erhält man Abbildung 3.



(a) Relativer Fehler beim Trapez-Leapfrog-Verfahren mit $w_H = 10$ und $w_L = 1$

(b) Relativer Fehler beim Adams-Bashforth-Verfahren mit $w_H = 10$ und $w_L = 1$

Abbildung 3: Entwicklung des relativen Fehlers abhängig von der Anzahl der Zeitschritte für das Trapez-Leapfrog-Verfahren und das Adams-Bashforth-Verfahren

Teil (a) von Abbildung 3 zeigt die Entwicklung des relativen Fehlers bei der Anwendung des Trapez-Leapfrog-Verfahrens zur Lösung der SWFW-Gleichung für den Spezialfall $w_H = 10$ und $w_L = 1$. Wählt man 10 Zeitschritte im Intervall $[0;1]$, so liegt der relative Fehler bei circa 1. Das Verfahren ist für diese zu große Schrittweite unbrauchbar. Bei Reduzierung der Schrittweite auf 0.01, was einer Zeitschrittanzahl von 100 entspricht, reduziert sich der relative Fehler

[16] Dies ist äquivalent zur Reduzierung der Zeitschrittgröße Δt.

auf circa 0.1. Es ist in Teil (a) ein linearer Zusammenhang zwischen relativem Fehler und der Anzahl der Zeitschritte zu erkennen. Der relative Fehler nimmt linear mit wachsender Anzahl an Zeitschritten ab. Jedoch lässt sich anhand dieser Grafik feststellen, dass zum Erhalt eines kleinen Fehlers – kleiner als 10^{-3} – ein enorm kleiner Zeitschritt gewählt werden muss.
Gut zu erkennen ist hier die Ordnung des Trapez-Leapfrog-Verfahrens. Dividiert man die Größe des Zeitschritts durch 10, so erhält man auch noch ein Zehntel des relativen Fehlers. Dies bestätigt, dass das Trapez-Leapfrog-Verfahren ein numerisches Verfahren erster Ordnung ist.

Generell gilt zur Bestimmung der Konvergenzordnung eines Verfahrens die Bedingung

$$\|relFehler\|_\infty \leq C \cdot \|\Delta t\|_\infty^p, \tag{16}$$

wobei bei Erfüllung dieser Gleichung für $C \geq 0$ die Variable p der Konvergenzordnung entspricht [22]. Setzt man nun die Werte resultierend aus Abbildung 3 in eine Umformulierung für Gleichung (16), die durch

$$p \approx \frac{|\, log(relFehler_i) \,| - |\, log(relFehler_{i+1}) \,|}{|\, log(\Delta t_i) \,| - |\, log(\Delta t_{i+1}) \,|} \tag{17}$$

gegeben ist, ein, so erhält man beim Trapez-Leapfrog-Verfahren als Ergebnis $p = 1$, was die Ordnung des Verfahrens widerspiegelt. Die Ordnung entspricht zudem der Steigung der Gerade in Abbildung 3. Bei einer oft geforderten Fehlergenauigkeit von 10^{-6} sind mehr als 10^7 Zeitschritte nötig.

In Teil (b) von Abbildung 3 ist der Verlauf des relativen Fehlers, der durch die Lösung der SWFW-Gleichung für $w_H = 10$ und $w_L = 1$ über dem Intervall $[0;1]$ mittels des Adams-Bashforth-Verfahrens entsteht, aufgetragen. Dieser beginnt ebenfalls für eine Schrittweite von 0.1 bei circa 1. Nach anfänglichen Konvergenzproblemen ist auch hier ein linearer Zusammenhang zwischen relativem Fehler und der Anzahl der Zeitschritte erkennbar. Jedoch hat die entstehende Gerade des relativen Fehlers beim Adams-Bashforth-Verfahren einen steileren Verlauf. Bei 100 Zeitschritten, was einer Schrittweite von 0.01 entspricht, liegt der relative Fehler zwar, genauso wie beim Trapez-Leapfrog-Verfahren, nahe bei 0.1, was jedoch aus den anfänglichen Konvergenzproblemen resultiert. Betrachtet man jedoch den weiteren Verlauf des relativen Fehlers des Adams-Bashforth-Verfahrens, so erkennt man, dass bei der Erhöhung der Anzahl der Zeitschritte um eine Größenordnung – von 100 auf 1000 – der relative Fehler um zwei Größenordnungen – von 0.1 auf 0.001 – reduziert wird. Daraus lässt sich schließen, dass das Adams-Bashforth-Verfahren ein numerisches Verfahren zweiter Ordnung ist. Nach der Startphase, die dies in der Grafik auf dem ersten Teil der Kurve ein wenig verfälscht, ergibt sich auch hier nach Gleichung (17) eine wahre Aussage für $p = 2$.
Da das Adams-Bashforth-Verfahren die Ordnung 2 besitzt, kann man mit ihm bereits mit weniger Zeitschritten bzw. einer größeren Schrittweite einen kleineren relativen Fehler erzielen. Doch auch hier benötigt man für eine Fehlergenauigkeit von 10^{-6} bereits $10^4 = 1000$ Zeitschritte.

Es ist anzufügen, dass beide Verfahren schnell und ohne großen Rechenaufwand diese Ergebnisse liefern. Doch schon diese kleine Auswahl an Qualitätsmerkmalen eines numerischen Lösers zeigt, dass das Trapez-Leapfrog-Verfahren und das Adams-Bashforth-Verfahren bei relativ geringem Aufwand nur ungefähre Lösungen der SWFW-Gleichung liefern. Der relative Fehler bleibt in beiden Fällen noch relativ groß.

Betrachtet man lineare Mehrschrittverfahren, wie z. B. die Adams-Verfahren, im Allgemeinen, so erkennt man, dass Verfahren höherer Ordnung nicht so einfach generiert werden können. Generell ist ein größerer Programmieraufwand erforderlich, um Mehrschrittverfahren von höherer Ordnung zu konstruieren. Die Koeffizienten für die allgemeine Formel der Mehrschrittverfahren müssen bestimmte Koeffizienten-Bedingungen erfüllen. Daher müssen sie für jede höhere Ordnung explizit berechnet bzw. implementiert werden [23].
Weiterhin weisen Mehrschrittverfahren bei wachsender Anzahl an Stufen Stabilitätsprobleme auf. Dahlquist hat bewiesen, dass es keine A-stabilen Mehrschrittverfahren mit m Schritten gibt, die eine Ordnung von $m > 2$ haben [18]. Diese wichtige Erkenntnis ist auch unter der 'Zweiten Dahlquist-Barriere' bekannt. Das bedeutet, dass es theoretisch möglich ist, ein Mehrschrittverfahren hoher Ordnung zu generieren, welches aber ab einer gewissen, hohen Ordnung aufgrund von Instabilitäten keine sinnvollen Ergebnisse mehr liefert. Darüber hinaus müssen für die Anwendung eines beliebigen Mehrschrittverfahrens mehrere Startwerte erzeugt werden. Sind diese nicht exakt berechenbar, so liefert jeder zusätzlich erforderliche Anfangswert ein zusätzliches Potential an Instabilität, da bereits diese Anfangswerte numerische Näherungen sind [24].

Eine numerische Technik, die auf genau die Problemstellung – einem hoch- und einem niedrigfrequenten Gleichungsanteil – hin entwickelt wurde, beruht auf sogenannten 'Splitting'-Methoden. 'Splitting' bedeutet in diesem Zusammenhang, dass die zu lösende Differentialgleichung in zwei Teile gespalten wird. Ein Teil enthält den hochfrequenten und der andere Teil den niedrigfrequenten Schwingungsanteil. Jeder Teil für sich erfordert eine andere Lösungstechnik: der langsame Anteil wird explizit und der schnelle Anteil implizit gelöst.

1.2.2. Operator-Splitting

Durch die zu Beginn dieses Kapitels erwähnte Aufspaltung der Differentialgleichung in zwei Summanden, kann für jeden Term separat ein numerisches Lösungsverfahren angewendet werden. Man kann somit eine Kombination aus dem einfachen, aber für diese Zwecke nicht stabilen expliziten und einem aufwendigen, aber stabilen impliziten Verfahren konstruieren. Damit kann die Rechenzeit zur Lösung der Differentialgleichung auf diese Weise minimiert werden, da der größte stabile Zeitschritt der semi-impliziten Diskretisierung wesentlich größer ist, als der größte stabile Zeitschritt für das explizite Verfahren [25]. Jedoch sollte es zu keinen Einbußen bezüglich der Genauigkeit der Lösung kommen. Um dies zu garantieren, müssen einige Aspekte bei der Modellbildung berücksichtigt werden. Einerseits müssen die schnellen Lösungskomponenten immer implizit behandelt werden. Andererseits sollte der implizite Gleichungsanteil leicht zu lösen sein, um dem Aufwand des semi-impliziten Verfahrens möglichst gering zu halten [25].

Bei der SWFW-Gleichung wird folglich der Term der schnellen Welle implizit und der Term der langsamen Welle explizit gelöst. Auf diese Weise werden die gerade aufgezählten Konditionen eingehalten und die Genauigkeit der Lösung soll durch die Verwendung eines semi-impliziten Verfahrens anstatt eines komplett impliziten Verfahrens möglichst nicht wesentlich reduziert werden [26].

Schon Klemp und Wilhelmson verfolgten 1978 diesen Ansatz und verwendeten bei der Diskretisierung des langsamen Gleichungsterms und des schnellen Gleichungsterms unterschiedliche Lösungstechniken, wie z. B. den Forward-Backward-Algorithmus für die schnellen Wellenterme und das Leapfrog-Verfahren für die langsamen Wellenterme [27]. Viele Jahre später, 1992, veröffentlichten Skamarock und Klemp eine Ausarbeitung über 'split-explicit schemes' zur Dämpfung von akustischen Wellen [28]. Daraufhin folgte 1998 von Wicker und Skamarock die Verbesserung der Splitting-Algorithmen durch Verwendung des zweistufigen Runge-Kutta-Verfahrens zweiter Ordnung für niedrigfrequente Anteile anstelle des Leapfrog-Verfahrens weiterhin in Kombination mit dem Forward-Backward-Algorithmus für die schnellen Wellen [29]. Erneut vier Jahre später analysierten die beiden anstelle des zweistufigen Runge-Kutta-Verfahrens ein dreistufiges Runge-Kutta-Verfahren in gleicher Funktion [14].
Kar kombinierte 2006 das dreistufige Runge-Kutta-Verfahren für schnelle Wellen anstelle des in den Anfängen genutzten Forward-Backward-Algorithmus [12]. Letztendlich wurde im Jahr 2007 das dreistufige Runge-Kutta-Verfahren zur Lösung der langsamen Anteile sogar für nichtlineare Probleme genutzt und in diesem Bereich analysiert [30].
Ein Nachteil der expliziten Runge-Kutta-Verfahren[17] ist jedoch, dass die hochfrequenten Wellen immer noch einen sehr kleinen Zeitschritt in den Lösungsverfahren benötigen, sofern kein zusätzlicher Dämpfungsterm eingeführt wird. Möchte man auf diesen künstlichen Dämpfungsterm verzichten, so muss auf andere Lösungsvarianten zurückgegriffen werden [10].

Es stellen sich die Fragen:

- Gibt es ein numerisches Lösungsverfahren für Differentialgleichungen, die verschiedene Dynamiken enthalten, welches die separate Behandlung der Gleichungsteile realisiert und aus diesem Grund stabile und genaue Ergebnisse für die atmosphärische Wissenschaft liefert?
- Wie sieht dieses Verfahren aus?
- Welche Kosten müssen für diese 'besseren' Lösungen in Kauf genommen werden?

Die Motivation dieser Arbeit liegt darin, ein neuartiges Verfahren im nachstehenden Kapitel von Grund auf zu erläutern und im weiteren Verlauf tiefer gehend zu analysieren für genau diesen Typus von Gleichung wie die SWFW-Gleichung. Somit wird ein Vergleich bezüglich des anfänglich zur Lösung benutzten Trapez-Leapfrog-Verfahrens bzw. Adams-Bashforth-Verfahrens ermöglicht.

[17] An dieser Stelle werden geteilte Methoden verwendet, welche explizite Runge-Kutta-Methoden für die langsame Dynamik mit einem beliebigen Verfahren für die schnelle Dynamik kombinieren.

Die vorliegende Masterarbeit umfasst also die Erklärung eines 'neuen' Verfahrens – Spectral Deferred Corrections, kurz SDC –, eine Kosten-Nutzen-Analyse gegen das Trapez-Leapfrog-Verfahren und das Adams-Bashforth-Verfahren und wird damit Antworten auf die gestellten Fragen geben.

2. Spectral Deferred Corrections

In diesem Kapitel werden die mathematischen Grundzüge des numerischen Verfahrens Spectral Deferred Corrections zur Lösung gewöhnlicher Differentialgleichungen, die in Form von Anfangswertproblemen gegeben sind, dargestellt.

Spectral Deferred Corrections – entwickelt im Jahr 2000 von Dutt et al. [31] – findet man in vielen verschiedenen Anwendungsgebieten als effizientes Lösungsverfahren von Modellgleichungen. Ein Einsatzgebiet ist das Lösen von Modellgleichungen, die auf der Shallow-Water-Gleichung aufsetzen, wie im Artikel von Jia, Hill, Evans, Fann, Taylor aus dem Jahr 2013 beschrieben wird [32].
Ebenfalls findet man Spectral Deferred Corrections[18] auch als Lösungsalgorithmus für Modelle, die auf Navier-Stokes-Gleichungen basieren. Aus Navier-Stokes-Gleichungen und Maxwell-Gleichungen setzt sich z. B. ein Modell zusammen, welches elektrisch leitende inkompressible Flüsse in der Gegenwart eines Magnetfelds simuliert [33]. Auch Michael Minion untersucht SDC angewandt auf inkompressible Navier-Stokes-Gleichungen mit Hilfe einer Approximation durch Boussinesq Konvektion [34].
SDC zusammen mit dem unstetigen Galerkin-Verfahren findet man auch in der Literatur zur Lösung von dynamischen Flussgleichungen [35] und in der mechanischen Modellierung von Wasser-Ressourcen-Systemen [36]. Darüber hinaus findet dieses Lösungsverfahren Anwendungen zur Modellierung kinetischer Systeme. Ein Beispiel ist die Modellierung eines kinetischen Modells für Flüssigkristallpolymere, wie es in der Veröffentlichung von Forest, Wang, Zhou von 2013 nachzulesen ist [37].
Wie in den obigen Referenzen, wird SDC nicht nur für gewöhnliche Differentialgleichungen genutzt. Das Verfahren kann mit Hilfe der Linienmethode[19] zur Lösung von partiellen Differentialgleichungen erweitert werden. Ein wichtiges Anwendungsbeispiel in diesem Zusammenhang und im Zusammenhang mit der SWFW-Gleichung ist die Lösung von Acoustic-Advection Systemen. An dieser Stelle wird jedoch nur auf diesen Bereich mittels Minion [34] verwiesen. Im Ausblick, Kapitel 5.2, erfolgen weitere Ausführungen hinsichtlich dieses Aspekts.
Weiterhin wird durch Xin, Huang, Zhao und Zhu im Artikel vom Jahr 2013 SDC als Lösungsverfahren für fraktale Differentialgleichungen analysiert [38].

Die Abkürzung SDC steht für Spectral Deferred Corrections. Die Funktionsweise dieser Methode wird durch die Namensgebung verdeutlicht.

1. Spectral:
   Spektral bezieht sich in diesem Fall auf die Einteilung des zu betrachtenden Zeitintervalls in mehrere Unterintervalle. Die Abstände der einzelnen Auswertungszeitpunkte innerhalb des großen Zeitintervalls müssen im Allgemeinen nicht äquidistant sein. Genutzt wird bei SDC eine spezielle numerische Integrationstechnik zur Lösung der Dif-

[18] Im Folgenden wird 'Spectral Deferred Corrections' per 'SDC' abgekürzt.
[19] Als Abkürzung findet man oft 'MOL', was dem englischen Ausdruck 'Method Of Lines' entspricht und mit Linienmethode ins Deutsche übersetzt wird.

ferentialgleichung bzw. des Integrals[20]: die Gauß-Lobatto-Quadratur.[21]
Auf die genaue Form der Anwendung und Funktionsweise dieser numerischen Quadratur wird in Kapitel 2.2 genauer eingegangen.

2. Deferred:
Übersetzt bedeutet dieses Adjektiv 'aufgeschoben' bzw. 'verzögert'. Es bezieht sich auf das ihm folgende Wort und wird daher im folgenden Punkt der Aufzählung im Zusammenhang mit 'Corrections' erklärt.

3. Corrections:
Charakteristisch für das iterative numerische Verfahren SDC ist die indirekte und schrittweise Verbesserung der berechneten Lösung. Allgemein kann man sich unter 'verzögerter Korrektur' folgendes vorstellen.
Zuerst wird über einem bestimmten Zeitgitter mit einem ausgewählten numerischen Verfahren eine Näherungslösung des Anfangswertproblems bestimmt. Daraufhin wird ein Korrekturterm berechnet, der sich aus der Differenz von exakter Lösung und genäherter Lösung ergibt. Auf diese Weise wird eine Korrektur der einfachen numerischen Lösung durch Addition des Korrekturterms ermöglicht. Wie dies im Detail funktioniert zeigt Kapitel 2.1.[22]

2.1. Mathematische Grundlagen für SDC

Die erste Veröffentlichung zur Beschreibung und theoretischen Herleitung von Spectral Deferred Corrections angewandt auf gewöhnliche Differentialgleichungen erfolgte im Jahr 2000 durch Dutt, Greengard und Rokhlin [31]. Neben den mathematischen und numerischen Grundlagen des Lösungsverfahrens werden für die Verwendung von expliziten, impliziten und kombinierten Euler-Verfahren als Basisverfahren schon erste Genauigkeits- und Stabilitätsanalysen durchgeführt. In Kapitel 3 der vorliegenden Arbeit wird auf genau diese Ergebnisse von Dutt et al. Bezug genommen.
Eine Arbeit, die darauf aufbauend verschiedene Quadraturformeln innerhalb des SDC-Verfahrens untersucht, ist im Jahr 2005 von Layton und Minion veröffentlicht worden [42].
Auf dem zentralen Artikel von Dutt et al. basieren viele weitere wissenschaftliche Arbeiten, die die Theorie von SDC darüber hinaus analysieren, in gewissen Aspekten erweitern und aus anderen Gesichtspunkten beschreiben. Eine Konvergenzanalyse im Vergleich mit anderen Methoden wurde z. B. von Hansen, Strain 2005 [43] und von Glaser, Rokhlin 2009 [44] durchgeführt. Verbesserungen des klassischen SDC-Verfahrens hinsichtlich der Konvergenz werden von Huang, Jia, Minion im Jahre 2006 vorgestellt [45].
Minion entwickelte einen neuen Aspekt des klassischen SDC-Verfahrens und publizierte diese Technik unter dem Namen 'Semi-Implicit Spectral Deferred Correction Methods for Ordinary

[20] Es wird die zum Anfangswertproblem äquivalente Volterra-Integral-Darstellung genutzt.
[21] Es können auch andere numerische Integrationsverfahren genutzt werden, die alle situationsbedingt Vor- und Nachteile aufweisen [39, 40].
[22] Allgemeine Informationen zu Deferred Corrections Methoden, unabhängig von SDC, sind in [41] nachzulesen.

Differential Equations' [26]. Dieser Aspekt ist für die vorliegende Arbeit von besonderer Bedeutung. Daher wird dieser Lösungsansatz in der Theorie in dem Unterkapitel 2.2.1 dargestellt und findet später Anwendung in der Lösung der SWFW-Gleichung.

Wie schon in der Einleitung zu Kapitel 2 erwähnt, ist die SDC-Methode zur Lösung von Anfangswertproblemen der Form

$$\phi'(t) = F(t,\phi(t)) \quad t \in [a,b] \tag{18}$$
$$\phi(a) = \phi_a \tag{19}$$

geeignet. Die Gleichung (18) zusammen mit der Gleichung (19) bilden somit die Form, in der die zu behandelnden Probleme vorliegen. Dabei ist t die in positive Richtung laufende Zeit, die von einer Startzeit a durch das Intervall hin zur Endzeit b fortschreitet. Es ist eine Differentialgleichung mit einer glatten Rechte-Seite-Funktion F und ein Anfangswert zum Zeitpunkt a gegeben. Die Differentialgleichung wird dadurch charakterisiert, dass die Ableitung $\phi'(t)$ von der Funktion selber, also von $\phi(t)$, funktional abhängt.
Die Integration beider Seiten der Differentialgleichung über den Zeitraum vom Startzeitpunkt a bis zur aktuellen Zeit t liefert die Gleichung

$$\int_a^t \phi'(s)ds = \int_a^t F(s,\phi(s))ds. \tag{20}$$

Mit dem Hauptsatz der Differential- und Integralrechnung erhält man

$$\phi(t) - \phi(a) = \int_a^t F(s,\phi(s))ds \tag{21}$$

als Zwischenergebnis und schließlich ergibt sich

$$\phi(t) = \phi_a + \int_a^t F(s,\phi(s))ds, \tag{22}$$

die sogenannte Volterra-Integralgleichung.[23]

Diese ist äquivalent zum Anfangswertproblem und folglich die Lösung der Differentialgleichung, welche die exakte, gesuchte Lösung ist. Die Äquivalenzbeziehung gilt nur unter der Voraussetzung, dass die Funktion F stetig ist und somit auch der Integrand auf der rechten Seite des Gleichheitszeichens stetig differenzierbar ist [47].

[23] Integralgleichungen sind vom Typ 'Volterra', wenn eine Integrationsgrenze fix und die andere Integrationsgrenze von einer freien Variable abhängt. Abgegrenzt dazu gibt es die sogenannten Integralgleichungen vom Fredholmschen Typ, in der beide Integrationsgrenzen fix sind [46].

SDC setzt nun auf dieser Integralgleichung auf. Durch jede Iteration des numerischen Lösungsverfahrens entsteht eine Näherung $\widetilde{\phi}(t)$ der exakten Lösung $\phi(t)$. Diese wird bei SDC mit einem Verfahren von geringer Ordnung berechnet. Der Fehler, der somit durch einen Approximationsschritt zum Zeitpunkt t entsteht, ist durch

$$\delta(t) = \phi(t) - \widetilde{\phi}(t) \tag{23}$$

gegeben und wird inhärent durch die Deferred Corrections korrigiert.[24]

Addiert man auf beiden Seiten der Gleichung $\widetilde{\phi}(t)$, so erhält man die Darstellung

$$\phi(t) = \delta(t) + \widetilde{\phi}(t) \tag{24}$$

für die exakte Lösung $\phi(t)$.

Wird dies nun in Gleichung (22) eingesetzt, ergibt sich

$$\delta(t) + \widetilde{\phi}(t) = \phi_a + \int_a^t F(s, \delta(s) + \widetilde{\phi}(s))ds, \tag{25}$$

wobei diese Gleichung zur sogenannten Korrekturgleichung umgeformt wird. Damit erschließt sich diese vorläufige Darstellung der Korrekturgleichung durch Subtraktion des Näherungsterms $\widetilde{\phi}(t)$ mit

$$\delta(t) = \phi_a + \int_a^t F(s, \delta(s) + \widetilde{\phi}(s))ds - \widetilde{\phi}(t). \tag{26}$$

Zusätzlich zu dieser Korrekturgleichung wird ein Fehlermaß

$$E(t, \widetilde{\phi}) = \phi_a + \int_a^t F(s, \widetilde{\phi}(s))ds - \widetilde{\phi}(t) \tag{27}$$

definiert. $E(t, \widetilde{\phi})$ ist ein Maß für den Fehler, der durch die Approximation $\widetilde{\phi}(t)$ entsteht. Dabei erfüllt $\widetilde{\phi}$ die Volterra-Gleichung (22) nicht. Ersetzt man $\phi(t)$ in Gleichung (22) durch $\widetilde{\phi}(t)$, so ergibt sich

$$\widetilde{\phi}(t) = \phi_a + \int_a^t F(s, \widetilde{\phi}(s))ds \tag{28}$$

als Volterra-Gleichung der Näherungslösung. Wie schon erwähnt ist diese Gleichung nicht erfüllt. Der Fehler, der in diesem Schritt auftritt, wird durch das Fehlermaß $E(t, \widetilde{\phi})$ beschrieben.

[24] Ein zweiter Ansatz an solche Probleme heranzugehen, ist die Verwendung von Diskretisierungs-Schemata von hoher Ordnung wie dies z. B. bei Runge-Kutta-Methoden oder linearen Mehrschritt-Methoden der Fall ist [31].

In das Fehlermaß geht daher vor allem der sogenannte Verstärkungsfehler ein. Er ist ein Indikator für die Stärke der Reaktion des numerischen Verfahrens auf kleine Änderungen der Funktionsparameter [19]. Die Funktion F bekommt hier die Näherungslösung übergeben. Dadurch entstehen Änderungen im Input, wobei das Fehlermaß dann die Änderungen im Output charakterisiert.
Wird die Gleichung (27) nach dem Anfangswert ϕ_a aufgelöst und in die Korrekturgleichung (26) eingesetzt, erhält man nach elementaren Äquivalenzumformungen und der Nutzung von Gleichung (24) die abschließende Korrekturgleichung

$$\delta(t) = \int_a^t F(s, \widetilde{\phi}(s) + \delta(s)) - F(s, \widetilde{\phi}(s))ds + E(t, \widetilde{\phi}), \tag{29}$$

siehe [26]. Diese Gleichung ist ein Indikator dafür, wie gut das Fehlermaß $E(t, \widetilde{\phi})$ den Wert $\delta(t)$ approximiert.
Die Vorgehensweise von SDC basiert nun auf dieser Korrekturgleichung. Man berechnet mit einem einfachen numerischen Verfahren eine Näherungslösung $\widetilde{\phi}(t)$ für das gegebene Anfangswertproblem.[25] Diese Lösung ist jedoch nur vorläufig, da die Genauigkeit dieser Lösung mit Hilfe der Korrekturgleichung (29), die auf demselben Gitter gelöst wird wie die approximative Lösung, verbessert wird. Um eine Reihe von Korrekturgleichungen zur Steigerung der Genauigkeit der Lösung zu erhalten, muss das Fehlermaß $E(t, \widetilde{\phi})$ mittels numerischer Quadratur approximiert werden [26].

Bei SDC verzichtet man darauf das erste Integral auch mit hoher Ordnung zu approximieren, denn ansonsten wird das Verfahren sehr teuer. Zu suchen ist ein Kompromiss zwischen Ordnung und Aufwand. Daher wird ein numerisches Lösungsverfahren mit niedriger Ordnung auf das erste Integral der Korrekturgleichung angewendet.

$$\int_a^t F(s, \widetilde{\phi}(s) + \delta(s)) - F(s, \widetilde{\phi}(s))ds \tag{30}$$

ist ein Integral über eine Differenz und drückt somit eine Änderung aus. Daher sollte bei dessen Auswertung ein Verfahren niedriger Ordnung ausreichend sein. Nach der Bestimmung der ersten Näherungslösung wird dann in der letzten Iteration das Fehlermaß bestimmt. Dabei versucht man möglichst hohe Ordnung zu erzielen, indem zur Näherung des Integrals die numerische Quadratur genutzt wird. Dies verursacht zwar generell einen größeren Aufwand, aber da hier $E(t, \widetilde{\phi})$ unabhängig von δ ist, fällt dieser nicht so sehr ins Gewicht wie bei der Lösung des ersten Integrals.

Zusammenfassend kann also mit einem numerischen Verfahren erster Ordnung zur Lösung von Differentialgleichungen, wie z. B. dem Euler-Verfahren, durch das Lösen einer Folge von

[25] Als einfache Lösungsalgorithmen werden oft das explizite oder implizite Euler-Verfahren genommen oder eine Kombination aus beiden Varianten.

Korrekturgleichungen und der damit verbundenen Korrektur eine wesentlich höhere Konvergenzordnung erreicht werden [31].

Das Unterkapitel 2.2 beschäftigt sich mit der Diskretisierung von SDC, um das Verfahren schließlich implementieren und effizient zum Lösen von Anfangswertproblemen nutzen zu können. Dabei ist der entscheidende Punkt eine geschickte Diskretisierung des Integrals innerhalb des Fehlermaßes der Korrekturgleichung.
Es folgt zunächst eine genaue Analyse des Diskretisierungsverfahrens. Im Anschluss daran liefert das Unterkapitel 2.3 einen entscheidenden Nachtrag zur Diskretisierung bezüglich der numerischen Integration.

2.2. Diskretisierung des Verfahrens

Um das in Kapitel 2.1 hergeleitete Verfahren implementieren zu können, muss dieses nun diskretisiert werden. Die zentrale Gleichung – charakteristisch für Deferred Corrections Methoden – ist durch

$$\phi^{k+1} = \phi^k + \delta^k, \tag{31}$$

die diskrete Darstellung von Gleichung (24), gegeben. k ist die Iterationsvariable, durch deren Inkrementierung die Verbesserung pro Iteration durchgeführt wird. Das bedeutet, dass die Lösung ϕ^k durch Summation mit dem Korrekturterm δ^k aus Schritt k verbessert wird und damit im $k+1$-ten Iterationsschritt eine genauere Lösung der zu lösenden Differentialgleichung entsteht.

Zur Diskretisierung von Gleichung (31), vor allem zur Diskretisierung von $\delta(t)$, wird eine diskrete Darstellung des in Gleichung (29) auftretenden Integrals benötigt. Hier kommt die numerische Integration – auch numerische Quadratur genannt –, die in dem nachstehenden Kapitel 2.3 genauer erläutert wird, ins Spiel. Bei der Implementierung von SDC im Rahmen dieser Arbeit wird ein spezielles Integrationsverfahren verwendet, nämlich die Gauß-Lobatto-Quadratur.
Das Gauß-Verfahren ist bekanntlich ein Kollokationsverfahren, wobei als Kollokationspunkte die Knoten der Gauß-Legendre-Quadratur gewählt werden. Diese werden im Unterkapitel 2.3.2 einmal beispielhaft berechnet.
Bei SDC soll die Lösung jedoch auf einem vorgegebenen Zeitintervall bestimmt werden. Vorgegeben sind also die Startzeit und die Endzeit. Würde nun die Standard-Gauß-Quadratur zur Integralapproximation genutzt, so wären die Randpunkte ebenfalls spektral angeordnet und entsprächen nicht den vorgegebenen fixen Intervallgrenzen.[26] Um die rechte und linke Intervallgrenze direkt miteinzubeziehen, wird hier die Gauß-Lobatto-Quadratur genutzt, welche im Unterkapitel 2.3.3 vorgestellt und im Unterkapitel 2.4.2 auf eine besonders effiziente Weise implementiert wird. Die Gauß-Lobatto-Quadratur ist in diesem Fall einfacher und etwas günstiger in der Implementierung, da der erste Schritt vom linken Rand zum ersten Knoten und außerdem die Auswertung der diskreten Volterra-Gleichung für den rechten Rand ausgespart

[26] Eine Möglichkeit wäre hier eine Extrapolation, um die Lösung am geforderten Endzeitpunkt zu erhalten [26].

werden kann [42]. Da bei dieser Form der Gauß-Quadratur der Anfangs- und Endkollokationspunkt fix sind und nicht mehr spektral im Gitter angeordnet, kommt es zu einer Reduktion der Konvergenzordnung [24]. Allerdings ist die höhere Ordnung des reinen Gauß-Verfahrens, vergleiche dazu Unterkapitel 2.3.3, in der Praxis von nicht so großer Bedeutung.

Bei der nun anschließenden Diskretisierung des SDC-Verfahrens werden Integrale der Form $\int_{\tau_m}^{\tau_{m+1}}$, welche z. B. in dem Ausdruck für $\delta(t)$ vorkommen, durch $I_m^{m+1}(\phi^k)$ per numerischer, interpolatorischer Quadratur approximiert. Dazu werden die Quadraturgewichte σ_m^l verwendet. Zusammen mit t_l, einer Zwischenstelle des Zeitintervalls $[t_n, t_{n+1}]$, und der Definition von $\phi_l^k \approx \widetilde{\phi}(t_l)$ wird nun die aufwendige Integration durch eine einfache Matrix-Vektor-Multiplikation der Form

$$\int_{\tau_m}^{\tau_{m+1}} F(s, \widetilde{\phi}(s))ds = \sum_{l=0}^{M} \sigma_m^l F(t_l, \phi_l^k) = I_m^{m+1}(\phi^k) \tag{32}$$

ersetzt. Diese Quadratur muss für jedes Unterintervall $[\tau_m; \tau_{m+1}]$ realisiert werden. Somit ist die Anzahl der Knoten bzw. Kollokationsstellen durch $M+1$ gegeben und m liegt im Bereich von 0 bis $M-1$ [26]. Die Funktion $F(t_l, \phi_l^k)$ muss also an $M+1$ Gauß-Lobatto-Quadratur-Knoten bekannt sein [26]. Die Matrix der Quadraturgewichte – σ_m^l – kann unabhängig vom Anfangswertproblem vor Ausführung des eigentlichen Verfahrens berechnet werden. Auf eine geschickte Berechnung der Gewichte σ_m^l wird in Unterkapitel 2.4.2 genauer eingegangen.
I_m^{m+1} ist also die exakte Lösung des Integrals des Lagrange-Interpolationspolynoms vom Grad M auf $M+1$ Knoten, was der Approximation per numerischer Quadratur von

$$\int_{\tau_m}^{\tau_{m+1}} F(s, \widetilde{\phi}(s))ds \tag{33}$$

entspricht.

SDC nutzt zur Lösung des ersten Integrals (30) einfache numerische Basis-Verfahren, da diese ausreichen um hohe Konvergenzordnungen zu erhalten [31]. Beispielhaft wird SDC im Folgenden für ein einfaches Standardverfahren notiert. Zunächst wird die Korrekturgleichung (29) durch die rechtsseitige Rechteckregel genähert,[27] so dass man auf

$$\delta_{m+1}^k = \delta_m^k + \Delta\tau_m[F(\tau_{m+1}, \phi_{m+1}^k + \delta_{m+1}^k) - F(\tau_{m+1}, \phi_{m+1}^k)] + E_{m+1}(\phi^k) - E_m(\phi^k) \tag{34}$$

kommt [26]. Die Implizitheit dieses Verfahrens wird in den Funktionsauswertungen F ersichtlich. Ebenfalls muss eine Diskretisierung des Fehlermaßes (27) erfolgen. E wird durch

$$E_{m+1}(\phi^k) - E_m(\phi^k) = I_m^{m+1}(\phi^k) - \phi_{m+1}^k + \phi_m^k \tag{35}$$

diskret dargestellt. Hier wird die im weiteren Verlauf der Arbeit noch genauer erläuterte Gauß-Lobatto-Quadratur benutzt.

[27] Man kann auch die linksseitige Rechteckregel oder eine Kombination aus links- und rechtsseitiger Rechteckregel als Basis-Verfahren nutzen.

Nun werden die letzten beiden Summanden aus Gleichung (34) durch den entsprechenden Ausdruck von Gleichung (35) ersetzt, so dass sich

$$\delta_{m+1}^k = \underline{\delta_m^k} + \Delta\tau_m[F(\tau_{m+1}, \phi_{m+1}^k + \delta_{m+1}^k) - F(\tau_{m+1}, \phi_{m+1}^k)] + I_m^{m+1}(\phi^k) - \phi_{m+1}^k + \underline{\phi_m^k} \quad (36)$$

als neue Variante der Korrekturgleichung ergibt.
Weiterhin folgt aus den beiden unterstrichenen Termen und Gleichung (31), dass

$$\underline{\delta_{m+1}^k} = \phi_m^{k+1} + \Delta\tau_m[F(\tau_{m+1}, \overline{\phi_{m+1}^k} + \overline{\delta_{m+1}^k}) - F(\tau_{m+1}, \phi_{m+1}^k)] + I_m^{m+1}(\phi^k) - \underline{\phi_{m+1}^k} \quad (37)$$

ist. Fasst man hier erneut jeweils die beiden markierten Terme zusammen und verwendet die Beziehung aus Gleichung (31), so ergibt sich

$$\phi_{m+1}^{k+1} = \phi_m^{k+1} + \Delta\tau_m[F(\tau_{m+1}, \phi_{m+1}^{k+1}) - F(\tau_{m+1}, \phi_{m+1}^k)] + I_m^{m+1}(\phi^k), \quad (38)$$

die finale diskrete Gleichung zur Implementierung von SDC mit der rechtsseitigen Rechteckregel als Basis-Verfahren [26].

2.2.1. Semi-implizite Diskretisierung

In der Diskretisierung aus Unterkapitel 2.2 wird als ein mögliches, simples Basis-Verfahren die rechtsseitige Implementierungsvariante der Rechteckregel genutzt. Ebenso könnte durch kleine Änderungen in der Diskretisierung die linksseitige Rechteckregel verwendet werden. In dieser Arbeit wird jedoch ein Schwerpunkt auf eine Kombination aus linksseitiger und rechtsseitiger Rechteckregel als Basis-Verfahren für SDC gelegt. Im späteren Verlauf der Arbeit stellt sich heraus, dass aus dieser Kombination der Basis-Verfahren das semi-implizite Euler-Verfahren resultiert.

Spectral Deferred Corrections für gewöhnliche Differentialgleichungen in der semi-impliziten Variante wurde 2003 von Michael Minion theoretisch hergeleitet und untersucht [26]. Als numerisches Beispiel betrachtete er neben seinen allgemeinen Genauigkeits- und Stabilitätsanalysen Van der Pol's Gleichung.
Weitere Untersuchungen bezüglich der semi-impliziten Diskretisierung von SDC stellten Hagstrom und Zhou im Jahr 2006 an [48].
Auch Minion vertiefte seine 2003 herausgebrachte Arbeit und untersuchte mit seiner neuen SDC-Variante inkompressible Flüsse, kombinierte SDC mit Runge-Kutta-Methoden und betrachtete Operator-Splitting angewandt auf physikalische Prozesse wie Advektion, Diffusion und Reaktion [34].

Die Diskretisierung der semi-impliziten Variante von SDC bedarf nur einiger kleinen Änderungen. Der zentrale Punkt ist die Aufspaltung der Differentialgleichung in eine Summe von zwei Termen. Diese Terme haben sehr unterschiedliche Eigenschaften. Man zerlegt die Differentialgleichung in eine Summe aus einer schnellen und einer langsamen Dynamik und behandelt einen Term implizit und den anderen Term explizit.[28]

[28] Nicht jede Differentialgleichung kann man als solch eine Summe auffassen.

Es wird somit das Anfangswertproblem

$$\phi'(t) = F(t,\phi(t)) = F_E(t,\phi(t)) + F_I(t,\phi(t)) \quad (39)$$
$$\phi(a) = \phi_a \quad (40)$$

betrachtet [26]. Folglich muss in allen Gleichungen $F(t,\phi(t))$ durch die Summe aus explizitem und implizitem Anteil ersetzt werden. Aus der zu implementierenden Gleichung (38)

$$\phi_{m+1}^{k+1} = \phi_m^{k+1} + \Delta\tau_m[F(\tau_{m+1},\phi_{m+1}^{k+1}) - F(\tau_{m+1},\phi_{m+1}^{k})] + I_m^{m+1}(\phi^k)$$

für die rechtsseitige Rechteckregel lässt sich nun diese zu implementierende Gleichung

$$\phi_{m+1}^{k+1} = \phi_m^{k+1} + \Delta\tau_m[F_E(\tau_m,\phi_m^{k+1}) - F_E(\tau_m,\phi_m^{k}) + F_I(\tau_{m+1},\phi_{m+1}^{k+1}) - F_I(\tau_{m+1},\phi_{m+1}^{k})] + I_m^{m+1}(\phi^k) \quad (41)$$

für eine Kombination aus linksseitiger und rechtsseitiger Rechteckregel – also für das semi-implizite Euler-Verfahren – ableiten [26]. Zur Berechnung der Quadratur-Matrix I muss in der Gleichung (32) ebenfalls die Summe aus $F_E + F_I$ berücksichtigt werden.

2.2.2. Diskretisierung über verschiedene Zeitgitter

Mit der Aufgabenstellung bzw. dem gegebenen Anfangswertproblem ist ein zu analysierendes Zeitintervall vorgegeben. Wie gerade gesehen, nutzt SDC einfache numerische Verfahren über ein Intervall $[a;b]$, um die Näherungslösung $\widetilde{\phi}(t)$ zu berechnen. Neben der Näherungslösung wird auf jedem Teilintervall $[t_n;t_{n+1}]$ die Korrekturgleichung $\delta(t)$ berechnet, so dass in jedem Schritt die Genauigkeit der numerischen Approximation verbessert werden kann.

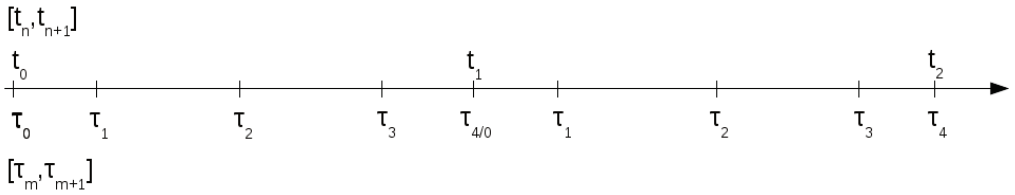


Abbildung 4: Intervalleinteilung bei SDC

Wie in Abbildung 4 zu erkennen ist, wird das gesamte Zeitintervall $[a;b]$ zunächst einmal in große Teilintervalle $[t_n;t_{n+1}]$ unterteilt. Diese werden wiederum in weitere Unterintervalle $[\tau_m;\tau_{m+1}]$ gespalten. Die τ_m besitzen relativ gesehen auf jedem großen Teilintervall die gleichen spektralen Abstände, die aus der Gauß-Lobatto-Quadratur resultieren.
Das bedeutet, dass bei der Implementierung diese zwei unterschiedlichen Zeitgitter betrachtet werden. Die τ_m sind die Nullstellen der Orthogonalpolynome – im Falle der vorliegenden SDC-Implementierung sind das die Nullstellen der Legendre-Polynome –, durch die das Lagrange-Interpolationspolynom gelegt wird. Somit wird iterativ für jedes Intervall $[t_n;t_{n+1}]$ eine Lösung per SDC-Algorithmus berechnet.

2.3. Numerische Integration

In diesem Unterkapitel wird nun auf die Diskretisierung des Fehlermaßes innerhalb der Korrekturgleichung genauer eingegangen.

Oft ist es nicht möglich, Integrale, wie sie in der Gleichung für das Fehlermaß $E(t,\widetilde{\phi})$ aus Gleichung (27) vorkommen, der Form

$$I(f) = \int_a^b f(x)dx, \tag{42}$$

für $f : [a;b] \to \mathbb{R}$, in geschlossener Form bzw. mit Hilfe einer Stammfunktion auszuwerten oder dieser analytische Lösungsalgorithmus würde zu großen Rechenaufwand erfordern. In diesen Fällen wird das Integral numerisch approximiert, da die exakte Lösung aus den soeben aufgeführten Gründen nicht analytisch berechnet werden kann [22].

In dem folgenden Abschnitt wird von dem Intervall $[a;b]$ als Ganzes ausgegangen. Bei einer Unterteilung in mehrere Unterintervalle sind die Grenzen der Unterintervalle jeweils wie a und b zu behandeln.
Eine einfache, intuitive Möglichkeit ist es, Stützstellen der Teilintervalle $x_0 < x_1 < \cdots < x_M$ mit $x_i \in [a;b]$ und $i \in [0;M]$ äquidistant zu wählen und ein Interpolationspolynom als Näherung für die zu integrierende Funktion f durch diese Stützstellen zu ermitteln. Dieses Vorgehen wird auch als interpolatorische Quadratur bezeichnet und basiert auf der Quadraturformel

$$I(f) \approx I_M(f) = (b-a)\sum_{m=0}^{M} \sigma_m f(x_m) \tag{43}$$

mit $\sigma_i \in \mathbb{R}$, wobei $I_M(f)$ möglichst nah an der exakten Lösung $I(f)$ liegen sollte [22]. Durch das Multiplizieren mit der Differenz aus oberer und unterer Intervallgrenze erfolgt eine Normierung auf den zu analysierenden Zeitbereich.

Die σ_m werden als Gewichte und die x_m als Knoten bzw. Stützstellen bezeichnet. Die einfachsten Möglichkeiten diese Formel auszuwerten führen auf die sogenannten Newton-Codes Regeln.[29]
Diese Herangehensweise nutzt zur Herleitung Lagrange-Polynome bzw. die Lagrange-Interpolation. Damit lässt sich die Näherung $I_M(f)$ mittels eines Interpolationspolynoms berechnen [22]. Bei gegebenen Knoten $x_0 < \cdots < x_M$ mit $x_i \in [a;b]$ werden die Gewichte σ_m aus Gleichung (43) durch das m-te Lagrange-Polynom bestimmt. Dabei ergibt

$$\sigma_m = \int_a^b L_m(t)dt \tag{44}$$

[29] Die bekanntesten sind hier die Rechteckregel, die Mittelpunktsregel, die Trapezregel, die Simpsonregel etc. Genaueres dazu ist in [49] nachzulesen.

zusammen mit

$$L_m(t) = \frac{\prod_{n \neq m}(t - x_n)}{\prod_{n \neq m}(x_m - x_n)} \tag{45}$$

das m-te Gewicht der interpolatorischen Quadraturformel. Das erhaltene Interpolationspolynom ist integrierbar und stellt eine Näherung für die eigentlich zu integrierende Funktion f dar [22]. Interessiert ist man nun an der möglichst besten Quadraturformel. Das bedeutet, dass die Lösung des exakten Integrals $I(f)$ möglichst nah an der Approximationslösung $I_M(f)$ liegen soll. Dazu betrachtet man als Quadraturfehler die Differenz aus exakter und approximierter Lösung

$$R_M(f) = I(f) - I_M(f).^{30} \tag{46}$$

Zur Fehlerminimierung gilt es die Knoten geschickt zu wählen. Da die Gewichte aus den gegebenen Knoten berechnet werden, stellen die Knoten bzw. deren Anordnung den einzigen Optimierungsparameter dar. Die Wahl der Knoten ist also der entscheidende Punkt, welcher die Qualität der sogenannten Gauß-Verfahren – auf die im Folgenden genauer eingegangen wird – entscheidend beeinflusst.

Es stellt sich heraus, dass die Wahl äquidistanter Stützstellen nicht empfehlenswert ist. Dies hat mehrere Gründe, die an dieser Stelle aufgezeigt werden.
Bei der Interpolation mit äquidistanten Knoten kann das sogenannte Runge-Phänomen auftreten. Am Rand des zu betrachtenden Intervalls kommt es bei der Verwendung eines Interpolationspolynoms hohen Grades zu Oszillationen. Das hat zur Folge, dass der Fehler bei Erhöhung des Polynomgrades nicht zwingend kleiner werden muss. Man kann sogar zeigen, dass der Approximationsfehler gegebenenfalls divergiert [51].[31]
SDC soll jedoch oft gerade mit Polynomen hohen Grades sehr gute Ergebnisse erzielen. Daher ist der soeben beschriebene Effekt des Runge-Phänomens zu vermeiden. Eine Möglichkeit diese Oszillationen am Rand des Intervalls zu minimieren, ist eine andere Wahl der Stützstellen, welche in den problematischen Randbereichen einen kleineren Abstand untereinander aufweisen. Hier gibt es mehrere Alternativen dies zu realisieren. Oft werden Tschebyscheff-Polynome verwendet [51]. Aber auch Gauß-Polynome besitzen die gerade erläuterte Eigenschaft, die zur Verbesserung des Approximationsfehlers führt.[32]

Die gerade erwähnten Gauß-Polynome haben noch eine weitere bedeutende Eigenschaft, weshalb man diese bei der Realisierung des SDC-Verfahrens bevorzugt. Unter der Verwendung eines Kollokationsverfahrens mit äquidistanten Stützstellen kann nur eine Konvergenzordnung von $p \leq M$ erreicht werden.[33] Sinnvoll ist daher eine sogenannte spektrale Anordnung der Knoten – wie bei dem klassischen Gauß-Ansatz –, womit ein optimales p/M-Verhältnis von $p = 2M$ erreicht werden kann [19]. Für alle Polynome $f(x)$ gilt bis zum Grad $2M-1$, dass das

[30] Genaueres dazu ist in der Literatur von Schaback und Wendland [50] zu finden.
[31] Ein ähnliches Phänomen tritt bei Fourierreihen unter der Bezeichnung Gibbsches Phänomen auf.
[32] Ein anderer verbreiteter Ansatz, die negativen Auswirkungen des Runge-Phänomens zu mindern, ist z. B. die Spline-Interpolation.
[33] p steht für die Konsistenzordnung und M für die Stufenzahl bzw. die Anzahl der Gewichte des Integrationsverfahrens.

approximierte Integral I_M gleich dem exakten Integral I bzw.

$$I_M(f) = \sum_{m=0}^{M} \sigma_m f(x_m) = \int_a^b f(x) \cdot w(x) dx = I(f) \tag{47}$$

ist. Um die unbekannten Variablen $x_0, x_1, \ldots, x_M$ und $\sigma_0, \sigma_1, \ldots, \sigma_M$ zu bestimmen, setzt man für $f(x)$ die Monome bis zum Grad $2M-1$, also $f(x) = x^k$ mit $k = 0, 1, \ldots, 2M-1$ ein. Gilt die Gleichung (47) für diese Monome exakt, dann gilt sie aufgrund der Linearität des Integrals für alle Polynome vom Grad kleiner oder gleich $2M-1$ exakt [50].
Man kann als Gewichtungsfunktion z. B. $w(x) = 1$ wählen. Dies ist genau jenes $w(x)$, bezüglich dem die Legendre-Polynome orthogonal im Sinne von

$$< f, g >= \int_a^b f \cdot g \cdot w(x) dx \tag{48}$$

sind. Dabei sind f und g beliebige Funktionen aus $L^2[a,b]$, dem Raum der zweifach Lebesgue-integrierbaren Funktionen [52]. Beispielhaft können so für $M = 2$ die Polynome auf dem Intervall $[-1;1]$ berechnet werden. In diesem Fall ist die Gewichtungsfunktion $w(x)$ unabhängig von x immer gleich 1. Zur Berechnung des Legendre-Polynoms dritter Ordnung werden diese vier Integrale

$$\int_{-1}^{1} 1 dx = 2 \tag{49}$$

$$\int_{-1}^{1} x dx = 0 \tag{50}$$

$$\int_{-1}^{1} x^2 dx = \frac{2}{3} \tag{51}$$

$$\int_{-1}^{1} x^3 dx = 0 \tag{52}$$

ausgewertet. Das daraus resultierende Gleichungssystem

$$\sigma_1 + \sigma_2 = 2 \tag{53}$$

$$\sigma_1 x_1 + \sigma_2 x_2 = 0 \tag{54}$$

$$\sigma_1 x_1^2 + \sigma_2 x_2^2 = \frac{2}{3} \tag{55}$$

$$\sigma_1 x_1^3 + \sigma_2 x_2^3 = 0 \tag{56}$$

erhält als Lösung $x_1 = -\frac{1}{\sqrt{3}}$ und $x_2 = \frac{1}{\sqrt{3}}$. Die zugehörigen Gewichte σ_1 und σ_2 sind beide identisch 1. Für alle Polynome bis zum Grad 3 ist die sich somit ergebende Quadraturformel

$$I_2 = f\left(-\frac{1}{\sqrt{3}}\right) + f\left(\frac{1}{\sqrt{3}}\right) \tag{57}$$

exakt.

Die Frage ist nun, ob es ein allgemeines Verfahren bzw. eine allgemeine Methode gibt, um die optimalen Knoten und Gewichte zu bestimmen, so dass

$$\int_a^b f(x) \cdot w(x)dx - \sum_{m=0}^{M} \sigma_m f(x_m) = 0 \tag{58}$$

ist. Dazu wird mit Hilfe der Stützstellen $x_0 < x_1 < \cdots < x_M$ ein Polynom

$$p(x) = \prod_{m=0}^{M} (x - x_m) \in \Pi_{M+1} \tag{59}$$

definiert.[34] Dieses Polynom hat eben diese Stützstellen x_m als Nullstellen [50]. Wählt man nun ein beliebiges Polynom $bp \in \Pi_{2M+1}$ und führt eine Division mit Rest von bp durch p durch, so kann bp durch

$$bp = r \cdot p + s \tag{60}$$

mit $r, s \in \Pi_M$ dargestellt werden. Geht man zudem von Erfüllung der Gleichung (47) auf Π_{2M+1} aus, dann gilt

$$\begin{aligned} I(bp) &= I(r \cdot p + s) && (61)\\ &= I_M(r \cdot p + s) && (62)\\ &= \sum_{m=0}^{M} \sigma_m (r(x_m)p(x_m) + s(x_m)) && (63)\\ &= \sum_{m=0}^{M} \sigma_m s(x_m) && (64)\\ &= I_M(s) && (65)\\ &= I(s). && (66) \end{aligned}$$

Damit dies erfüllt ist, muss

$$I(r \cdot p) = \int_a^b r(x)p(x) \cdot w(x)dx = 0 \tag{67}$$

für alle $r \in \Pi_M$ gelten [50]. Diese Bedingung wird im nachstehenden Unterkapitel 2.3.1 als Definition von Orthogonalität per Gleichung (68) eingeführt. Es bedeutet, dass die Stützstellen, über die p definiert ist, Nullstellen eines zu Π_M orthogonalen Polynoms vom Grad $M+1$ sein müssen [50]. Auf dieser Basis kann die Exaktheit der Quadraturformel auf den Raum Π_{2M+1} übertragen werden. Daraus ergibt sich eine sowohl notwendige als auch hinreichende Bedingung für die Konstruktion optimaler Stützstellen für die Interpolation [50].

Im nächsten Abschnitt wird genau diese Bedingung der Orthogonalität näher untersucht. Beispielhaft wird dort u. a. gezeigt, dass x_1 und x_2 die Nullstellen des sich zur Ordnung 2 ergebenden Legendre-Orthogonalpolynoms aus Unterkapitel 2.3.2 sind.
Es wird im Folgenden genauer auf diese spektrale Knotenwahl und Konstruktion durch Orthogonalpolynome eingegangen.

[34] Π_j ist der Raum aller reellen Polynome bis zum Grad j.

2.3.1. Orthogonalpolynome

Bei der optimalen Knotenwahl numerischer Integrationsverfahren spielen Orthogonalpolynome eine wichtige Rolle. Anstatt der trivialen Vorgehensweise, äquidistante Stützstellen zu verwenden, nutzt man die Nullstellen der Orthogonalpolynome als Knoten bei der Interpolation. In diesem Unterkapitel wird zunächst die Definition von Orthogonalpolynomen aufgeführt. Im Anschluss daran folgt die Verwendung dergleichen bei der numerischen Integration.

Orthogonalität wird über das Skalarprodukt definiert. Sei μ ein beliebiges Borel-Maß auf $\mathbb{R}$ und $L^2(\mathbb{R}, d\mu)$ ein Hilbertraum, dann ist das L^2-Skalarprodukt zweier Polynome $p, q \in \Pi$ durch

$$\langle p, q \rangle = \int_a^b p(x)q(x)\mu(dx) \tag{68}$$

definiert. Bei bekannter Gewichtungsfunktion $w(x) : [a;b] \to (0;\infty]$ kann das Borelsche Maß μ durch $\mu(dx) = w(x)dx$ ersetzt werden.
Das Skalarprodukt ist also eine Abbildung von $\Pi \times \Pi$ in die reellen Zahlen mit u. a. der Eigenschaft, dass

$$\langle p, p \rangle \geq 0 \text{ für } p \in \Pi \tag{69}$$

und

$$\langle p, p \rangle = 0 \Leftrightarrow p = 0 \tag{70}$$

ist. Man nennt p und q aus dem Raum aller reellen Polynome orthogonal zueinander, wenn das Skalarprodukt aus p und q Null ergibt. Diese zueinander orthogonalen Polynome werden dann auch Orthogonalpolynome genannt [22].
Zum Existenznachweis und zur Konstruktion einer speziellen Folge paarweiser orthogonaler Polynome kann die Gram-Schmidt-Orthogonalisierung der Momente x^n mit $n \in \mathbb{N}$ genutzt werden [22]. Das allgemeine Gram-Schmidt-Verfahren konstruiert auf der Basis $q_0, \ldots q_n$ eines beliebigen Unterraums U eine Orthonormalbasis von U [52].

Das Orthogonalisierungsverfahren nach Gram-Schmidt ist jedoch eine recht aufwendige Methode zur Berechnung von Orthogonalpolynomen. Aus diesem Grund werden Orthogonalpolynome in der Praxis meistens effizienter mittels einer Rekursionsformel, der so genannten Drei-Term-Rekursion bzw. dem Lanczos-Verfahren, berechnet. Diese kann aus dem allgemeinen Gram-Schmidt-Verfahren hergeleitet werden [22].
Durch die Wahl verschiedener Gewichtungsfunktionen $w(x)$ in Gleichung (68) können unterschiedliche Orthogonalpolynome erzeugt werden. In der Implementierung von SDC werden Legendre-Polynome mit zugehöriger Gewichtungsfunktion $w(x) = 1$ gewählt.[35]

Auf eine effiziente Implementierung zur Auswertung dieser Rekursionsformel wird in Kapitel 2.4.2 eingegangen.

[35] Andere Möglichkeiten wären Tschebyscheff-Polynome, Jacobi-Polynome, Hermite-Polynome, Laguerre-Polynome etc. [22].

2.3.2. Beispiel: Legendre-Polynome

Auf die Herleitung verschiedener Rekursionsformeln wird an dieser Stelle verzichtet. Anstatt dessen erfolgt die Anwendung der gegebenen Rekursionsvorschrift

$$p_0 = 1 \tag{71}$$
$$p_1 = x \tag{72}$$
$$p_{n+1} = \frac{2n+1}{n+1} \cdot x \cdot p_n - \frac{n}{n+1} \cdot p_{n-1} \tag{73}$$

mit $n \in \mathbb{N}$ am Beispiel der Legendre-Polynome [53].
p_2 und p_3 usw. lassen sich damit relativ leicht berechnen. Mit

$$p_2 = \frac{2 \cdot 1+1}{1+1} \cdot x \cdot x - \frac{1}{1+1} \cdot 1 = \frac{3}{2}x^2 - \frac{1}{2} \quad \text{und} \tag{74}$$
$$p_3 = \frac{2 \cdot 2+1}{2+1} \cdot x \cdot (\frac{3}{2}x^2 - \frac{1}{2}) - \frac{2}{2+1} \cdot x = \frac{5}{2}x^3 - \frac{3}{2}x \tag{75}$$

kommt man dann auf die gesuchten Nullstellen dieser speziellen Legendre-Orthogonalpolynome. Durch Nullsetzen von p_1 erhält man $x_1 = 0$ als erste Nullstelle. Für p_2 und p_3 erhält man auf gleiche Art und Weise $x_2 = \pm\sqrt{\frac{1}{3}}$ und $x_3 = \pm\sqrt{\frac{3}{5}}$ oder $x_3 = 0$.
Aus diesen Knoten können durch die Lagrange-Interpolation die der Ordnung entsprechenden Gewichte berechnet werden, wie in den Formeln (44) und (45) beschrieben wird.
Für das Lagrange-Polynom der Ordnung 2 zu $-\sqrt{\frac{1}{3}}$ mit

$$L_{2,1}(t) = \frac{t + \sqrt{\frac{1}{3}}}{\sqrt{\frac{1}{3}} + \sqrt{\frac{1}{3}}} \tag{76}$$

und auf gleiche Weise für das Lagrange-Polynom der Ordnung 2 zu $\sqrt{\frac{1}{3}}$ mit

$$L_{2,2}(t) = \frac{t - \sqrt{\frac{1}{3}}}{-\sqrt{\frac{1}{3}} - \sqrt{\frac{1}{3}}} \tag{77}$$

erhält man als zugehörige Gewichte

$$\sigma_{2,1} = \int_{-1}^{1} L_{2,1}(t)dt = 1 \tag{78}$$

und

$$\sigma_{2,2} = \int_{-1}^{1} L_{2,2}(t)dt = 1. \tag{79}$$

Interessanter wird es jedoch erst ab Ordnung 3. Dabei ist das Lagrange-Polynom für den Knoten $\sqrt{\frac{3}{5}}$ mit

$$L_3(t) = \frac{t + \sqrt{\frac{3}{5}}}{\sqrt{\frac{3}{5}} + \sqrt{\frac{3}{5}}} \cdot \frac{t+0}{\sqrt{\frac{3}{5}} + 0} \tag{80}$$

gegeben. Die daraus resultierenden Gewichte sind

$$\sigma_{3,1} = \int_{-1}^{1} L_3(t)dt = \frac{5}{9}. \tag{81}$$

$\sigma_{3,2} = \frac{8}{9}$ und $\sigma_{3,3} = \frac{5}{9}$ lassen sich ebenfalls mit Hilfe der Lagrange-Interpolationspolynome berechnen.
Für eine Gauß-Quadraturformel vom Grad n benötigt man genau die Nullstellen des n-ten Orthogonalpolynoms und die entsprechenden Gewichte [22].

Meist ist der erste Schritt, vor der eigentlichen numerischen Integration, eine Zerlegung des gesamten Intervalls in viele kleinere Intervalle, so dass $a = x_0 < x_1 < \cdots < x_M = b$ ist. Das Integral wird über jedes Teilintervall ausgewertet und die Ergebnisse werden aufsummiert. Also ist

$$I(f) = \int_a^b f(x)dx = \sum_{m=1}^{M} \int_{x_{m-1}}^{x_m} f(x)dx = \sum_{m=1}^{M} (x_m - x_{m-1}) \int_0^1 f(x_{m-1} + t \cdot (x_m - x_{m-1}))dt \tag{82}$$

die neue Gleichung, jedoch noch mit exaktem Ergebnis [22]. Die letzte Gleichheit ergibt sich aus einer Variablentransformation mit $x = x_{m-1} + t \cdot (x_m - x_{m-1})$.
Es bleibt jetzt noch das Problem

$$\int_0^1 g(t)dt \tag{83}$$

mit $g(t) = f(x_{m-1} + t \cdot (x_m - x_{m-1}))$ approximativ zu lösen [22].

2.3.3. Gauß-Lobatto-Quadratur

Allgemein erfolgt die Berechnung der Legendre-Orthogonalpolynome und deren Gewichte über dem Intervall $[-1;1]$, wie es auch in Kapitel 2.3.2 durchgeführt wurde. Bei SDC wird jedoch oft – aus Gründen der Einfachheit bzw. Unkompliziertheit – eine spezielle Form der Gauß-Quadratur und zwar die Gauß-Lobatto-Quadratur verwendet.
Die Besonderheit bei der Gauß-Lobatto-Quadratur ist, dass der Anfangs- und Endpunkt des zu betrachtenden Intervalls mit einbezogen und sozusagen fix gesetzt werden. Diese beiden Kollokationspunkte sind nicht spektral, was zur Folge hat, dass sich die Ordnung leicht verringert. Das allgemeine Gauß-Verfahren besitzt einen Genauigkeitsgrad von $2M$, wobei M die Anzahl der Stützstellen bezeichnet [23]. Das bedeutet, dass Polynome vom maximalen Grad $2M-1$ durch die Quadraturformel exakt integriert werden [23, 52]. Wohingegen beim Gauß-Lobatto-Verfahren nicht mehr als der Genauigkeitsgrad $2M-2$ erreicht werden kann [23]. Die Ordnung reduziert sich um zwei, nämlich durch die beiden fest vorgegebenen Intervallendpunkte. Auch hier ist die angegebene Ordnung äquivalent dazu, dass mit der Gauß-Lobatto-Quadraturformel Polynome bis zur Ordnung $2M-3$ exakt integriert werden können [52].

Man gehe von der allgemeinen Gauß-Lobatto-Quadratur auf dem Intervall $[-1;1]$ aus, die jedoch, wie im letzten Kapitel beschrieben, auf andere Intervalle umgerechnet werden kann. Das Integral wird durch

$$\int_{-1}^{1} f(x)dx \approx w_0 f(-1) + w_M f(1) + \sum_{i=1}^{M-1} w_i f(x_i) \tag{84}$$

approximiert, wobei die Knoten $x_0 = -1$ und $x_M = 1$ fix sind und sich die restlichen Knoten x_i für $1 \leq i \leq M-1$ aus den Nullstellen der Ableitung des Legendre-Orthogonalpolynoms p_M ergeben [54]. Für einen Knoten x_i gilt also in dem gerade genannten Intervall

$$p'_M(x_i) = 0. \tag{85}$$

Zusammen mit den beiden vorgegebenen Endpunkten -1 und 1 kann die vollständige Gauß-Lobatto-Quadraturformel aufgestellt werden. Die Gewichte

$$w_i = \frac{2}{M(M+1)[p_M(x_i)]^2} \tag{86}$$

für $0 \leq i \leq M$ eingesetzt in Gleichung (84) ergeben die gesuchte Quadraturformel [54]. Ergänzend ist anzufügen, dass die Knoten symmetrisch um das Zentrum angeordnet sind.

2.4. Implementierung von SDC

Zur Implementierung von Spectral Deferred Corrections wird die in Kapitel 2.2 hergeleitete diskrete Formel (38) genutzt, die hier zur Übersichtlichkeit noch einmal aufgeführt wird:

$$\phi_{m+1}^{k+1} = \phi_m^{k+1} + \Delta\tau_m[F(\tau_{m+1}, \phi_{m+1}^{k+1}) - F(\tau_{m+1}, \phi_{m+1}^{k})] + I_m^{m+1}(\phi^k) \tag{87}$$

Ebenfalls geht bei der Implementierung von $I_m^{m+1}(\phi^k)$ die in Kapitel 2.3.3 dargestellte Gauß-Lobatto-Quadratur ein.

2.4.1. Programmablauf

Der folgende Pseudocode zeigt den Kern des in dieser Arbeit implementierten und genutzten SDC-Programms, welches in Matlab geschrieben ist.

Algorithm 1 Iterativer SDC-Algorithmus

```
1:  procedure SDC(M, K, N, init, T, lambda)
2:      t ← T(1)
3:                          ▷ aktuelle Zeit t wird Anfangspunkt des gegebenen Zeitintervalls T
4:      te ← T(2)
5:                           ▷ maximale Zeit te wird Endpunkt des gegebenen Zeitintervalls T
6:      Δt ← te/N
7:      u_0 ← init
8:      knotenabstaende ← Gauß-Lobatto-Knoten mittels Eigenwerten berechnen
9:      gewichte ← per knotenabstaende berechnen

10:     for n ← 1, N do
11:                 ▷ n geht über die großen Zeitschritte t_n, in die das Intervall T geteilt wird
12:         t ← t + Δt
13:         f ← Testfunktion auswerten für aus letzter Iteration bekannten Teil bei geg. λ
14:         for k ← 1, K do
15:                                   ▷ Verbesserung der Lösung über Unter-Unterintervalle
16:             Quadratur-Matrix I ← Δt · (gewichte · f)
17:             u ← Startwert: u_0
18:             f ← Startwert: Testfunktion an u_0
19:             for m ← 1, M do
20:                                                ▷ kleine spektrale Schritte τ_m
21:                 u ← Lösung von Gleichung (87) auf spektral angeordneten Knoten
22:                 f ← Lösung der Testfunktion auf spektral angeordneten Knoten
23:             end for
24:         end for
25:         u_0 ← mit letztem Wert u(t_n) = u(τ_m) neu belegen
26:     end for
27: end procedure
```

Das SDC-Verfahren wird durch die Eingabe von sechs Parametern gesteuert. Die letzten drei Übergabeparameter sind mittels der Variable *init*, dem Anfangswert für das zu lösende Anfangswertproblem, mittels der Variable T, einem zu analysierenden Zeitintervall – in den meisten der dieser Arbeit zugrunde liegenden Rechnungen ist dies durch $T = [0;1]$ gegeben – und mittels der Variable *lambda*, dem entsprechenden λ aus dem vorliegenden Anfangswertproblem, zu belegen. Weiterhin sind bei jedem Programmaufruf die variablen Parameter M, K und N zu übergeben. Dies sind die Iterationsanzahlen für die drei ineinander geschachtelten For-Schleifen.
N ist die Anzahl der großen Zeitschritte, vergleiche mit den Zeitintervallen $[t_n; t_{n+1}]$ in Abbildung 4. Zudem kann die Zahl M, welche der Menge der Quadratur-Knoten entspricht, von außen vorgegeben werden. Diese entspricht der Anzahl der Intervalle $[\tau_m; \tau_{m+1}]$ von Abbildung 4. In der innersten For-Schleife finden die SDC charakterisierenden Korrekturschritte statt. Berechnet wird hier die Korrekturgleichung über den spektral angeordneten Knoten τ_m innerhalb der großen Zeitintervalle.

Die Ausführung der innersten Schleife findet jedoch nicht nur einmal statt, sondern sie wird k-mal iteriert. In jedem k-ten Schritt erfolgt eine Verbesserung der ursprünglichen numerischen Approximation durch die numerische Auswertung der Korrekturgleichung.
Aus der innersten For-Schleife über M können zwei Rechnungen in die For-Schleife über K ausgelagert werden. Wie in Gleichung (38) bzw. Gleichung (87) zu sehen ist, hängen die letzten beiden Terme der rechten Seite nur von k ab. Somit kann die Berechnung der Quadratur-Matrix I und eine Auswertung der Funktion F ausgelagert werden und über alle M Schritte der innersten Schleife beibehalten werden. Ist nun eine Lösung über das erste Intervall $[t_n;t_{n+1}]$ gefunden, wird diese als Anfangswert für das nächste große Zeitintervall verwendet.
Dies wird so lange wiederholt, bis eine Lösung für den kompletten zu analysierenden Zeitbereich gefunden wurde bzw. über alle großen Zeitintervalle $[t_n;t_{n+1}]$ iteriert wurde.

2.4.2. Implementierung der Gauß-Quadratur

In Kapitel 2.2 ist in Gleichung (32) bereits auf formale Art und Weise die numerische Quadratur zur Bestimmung einer Näherungslösung für das Integral eingeführt worden. Bei der Berechnung der Matrix $I_m^{m+1}(\phi^k)$ wird $F(t_l,\phi_l^k)$ mit Gewichten σ_m^l multipliziert. Diese Gewichte sind entscheidend für die Effizienz der Quadratur. Weiterhin wird in Kapitel 2.3.1 die Drei-Term-Rekursion als Verfahren zur Berechnung von Orthogonalpolynomen genannt. Auf deren Basis werden dann die Knoten und Gewichte für die entsprechende Gauß-Quadratur berechnet. Eine vorteilhafte, in SDC benutzte Berechnungsvariante der Quadratur-Knoten, die ebenfalls auf der Drei-Term-Rekursion aufsetzt, verwendet die Eigenwerte einer speziellen Tridiagonalmatrix [55].

Zunächst wird an dieser Stelle die gerade erwähnte Rekursionsbeziehung noch einmal mathematisch exakt durch

$$p_m(x) = (a_m x + b_m)\cdot p_{m-1}(x) - c_m p_{m-2}(x) \tag{88}$$

notiert. Die Gleichung (88) ist für $1 \leq m \leq M$ mit den Anfangswerten $p_{-1}(x) = 0$ und $p_0(x) = 1$ gegeben. Der Name Drei-Term-Rekursion stammt von der Verknüpfung der Orthogonalpolynome mit drei verschiedenen Indizes. Um p_m ausrechnen zu können, greift man rekursiv auf zwei zuvor berechnete Orthogonalpolynome p_{m-1} und p_{m-2} zurück. Aus diesem Grund ist auch die Vorgabe von zwei Anfangswerten erforderlich.
Die Gleichung (88) ist sehr allgemein gehalten und für beliebige Orthogonalpolynome anzuwenden. Dazu müssen je nach Polynomtyp die Koeffizienten a_m, b_m und c_m entsprechend gewählt werden. Für die bei SDC verwendeten Legendre-Polynome lautet die konkrete Variante zu Gleichung (88)

$$p_{m+1}(x) = \frac{2m+1}{m+1} x \cdot p_m(x) - \frac{m}{m+1} p_{m-1}(x) \tag{89}$$

für $m \in [1;M]$ mit $M \in \mathbb{N}$ [23]. Diese lässt sich mittels Substitution von m zu $m+1$ auf die erste rekursive Formel

$$p_m(x) = \frac{2m-1}{m} x \cdot p_{m-1}(x) - \frac{m-1}{m} p_{m-2}(x) \tag{90}$$

zurückrechnen. Der Parameter m steht in allen Fällen für das m-te Orthogonalpolynom. Zudem liegt m zwischen 0 und der gewünschten Anzahl verwendeter Quadratur-Knoten. Damit sind die gesuchten Koeffizienten der Drei-Term-Rekursion für Legendre-Polynome durch

$$a_m = \frac{2m-1}{m} \tag{91}$$

und

$$c_m = \frac{m-1}{m} \tag{92}$$

gegeben. Somit würden sich für fünf Quadratur-Knoten, also $0 \leq m \leq 5$, z. B. die Koeffizienten $a_1 = 1$, $a_2 = 1.5$, $a_3 = 1.6667$, $a_4 = 1.75$ und $a_5 = 1.8$ ergeben. Die Koeffizienten b_m sind bei den Legendre-Polynomen stets identisch Null.
Diese Drei-Term-Rekursionsvorschrift kann nun in Form von Matrizen und Vektoren mathematisch umformuliert werden.[36] Dazu wird die allgemeine Gleichung (88) nach $x \cdot p_{m-1}$ aufgelöst und man erhält

$$x \cdot p_{m-1}(x) = \frac{1}{a_m} p_m(x) - \frac{b_m}{a_m} p_{m-1}(x) + \frac{c_m}{a_m} p_{m-2}(x), \tag{93}$$

was demzufolge als Matrix-Gleichung für $m \in [1; M]$ die Form

$$x \begin{bmatrix} p_0(x) \\ p_1(x) \\ \vdots \\ \vdots \\ \vdots \\ p_{M-1}(x) \end{bmatrix} = \begin{bmatrix} -\frac{b_1}{a_1} & \frac{1}{a_1} & 0 & \cdots & \cdots & \cdots \\ \frac{c_2}{a_2} & -\frac{b_2}{a_2} & \frac{1}{a_2} & 0 & \cdots & \cdots \\ 0 & \ddots & \ddots & \ddots & 0 & \cdots \\ 0 & 0 & \ddots & \ddots & \ddots & \ddots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \frac{1}{a_{M-1}} \\ \vdots & \ddots & \ddots & \ddots & \frac{c_M}{a_M} & -\frac{b_M}{a_M} \end{bmatrix} \begin{bmatrix} p_0(x) \\ p_1(x) \\ \vdots \\ \vdots \\ \vdots \\ p_{M-1}(x) \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \vdots \\ \vdots \\ 0 \\ \frac{p_M(x)}{a_M} \end{bmatrix} \tag{94}$$

annimmt. Für die zweite Vektorkomponente des Vektors auf der linken Seite des Gleichheitszeichens ergibt sich z. B.

$$x \cdot p_1(x) = \frac{c_2}{a_2} p_0(x) - \frac{b_2}{a_2} p_1(x) + \frac{1}{a_2} p_2(x), \tag{95}$$

was identisch mit Gleichung (93) für $m = 2$ ist.
Um diese Matrix-Gleichung genauer zu untersuchen werden die Parameter

$$p := \begin{bmatrix} p_0(x) \\ p_1(x) \\ \vdots \\ \vdots \\ \vdots \\ p_{M-1}(x) \end{bmatrix}, \tag{96}$$

[36] Dies wird hier im allgemeinen Fall für beliebige Orthogonalpolynome gezeigt.

$$T := \begin{bmatrix} -\frac{b_1}{a_1} & \frac{1}{a_1} & 0 & \cdots & \cdots & \cdots \\ \frac{c_2}{a_2} & -\frac{b_2}{a_2} & \frac{1}{a_2} & 0 & \cdots & \cdots \\ 0 & \ddots & \ddots & \ddots & 0 & \cdots \\ 0 & 0 & \ddots & \ddots & \ddots & \ddots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \frac{1}{a_{M-1}} \\ \vdots & \ddots & \ddots & \ddots & \frac{c_M}{a_M} & -\frac{b_M}{a_M} \end{bmatrix} \tag{97}$$

und

$$r := \begin{bmatrix} 0 \\ 0 \\ \vdots \\ \vdots \\ 0 \\ \frac{p_M(x)}{a_M} \end{bmatrix} \tag{98}$$

definiert. Schreibt man nun die Matrix-Gleichung (94) mit Hilfe der gerade definierten Parameter p, T und r, so erhält man

$$x \cdot p = T \cdot p + r. \tag{99}$$

Bei der Suche der Nullstellen vom $p_M(x)$ ist die Gleichung (99) sehr hilfreich. Es gilt nämlich

$$x \cdot p(x) = T \cdot p(x) \qquad \Longleftrightarrow \qquad r = 0 \tag{100}$$

und mit $r = 0$ auch $p_M(x) = 0$. Damit beschränkt sich die Nullstellensuche des Orthogonalpolynoms $p_M(x)$ auf die Lösung des Eigenwertproblems. Die Nullstellen von $p_M(x)$ sind somit die Eigenwerte der Tridiagonal-Matrix T.[37] Es reicht also für $p \neq 0$ die Gleichung $x \cdot p = T \cdot p$ zu lösen, um die Nullstellen des Orthogonalpolynoms zu erhalten.
Falls eine Menge von orthonormalen Polynomen vorliegt, ist die Matrix T symmetrisch.[38]

Im Fall der Legendre-Interpolation ergibt sich die Matrix

$$T_{Leg} = \begin{bmatrix} 0 & 1 & 0 & 0 & \cdots & \cdots \\ \frac{1}{3} & 0 & \frac{2}{3} & 0 & \cdots & \cdots \\ 0 & \ddots & \ddots & \ddots & 0 & \cdots \\ 0 & 0 & \ddots & \ddots & \ddots & \ddots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \frac{M-1}{2M-3} \\ \vdots & \ddots & \ddots & \ddots & \frac{M-1}{2M-1} & 0 \end{bmatrix}, \tag{101}$$

[37] Die Matrix T bildet p auf ein Vielfaches von sich selbst ab. Somit ist x der zugehörige Eigenwert [22].

[38] Oft überführt man T zur effizienten und stabilen Berechnung der Eigenwerte mittels einer Ähnlichkeitstransformation in eine symmetrische Matrix. Die einfachste Möglichkeit dazu, wäre eine Skalierung mit einer Diagonalmatrix. Durch diese Ähnlichkeitstransformation bleiben die Eigenwerte unverändert [55]. Jedoch lassen sich Eigenwertprobleme symmetrischer Matrizen numerisch besser lösen. Dazu kann z. B. der sogenannte Q-R-Algorithmus genutzt werden [55].

welche offensichtlich nicht symmetrisch ist. Das liegt daran, dass durch die Rekursionsvorschrift eine Menge von orthogonalen Polynomen erzeugt wird, die jedoch nicht normiert sind.

Die in Gleichung (97) definierte Matrix T kann in einem speziellen Fall leicht zur Berechnung der Gauß-Lobatto-Knoten modifiziert werden. Dazu werden besondere Eigenschaften der Legendre-Polynome im Intervall $[-1;1]$ genutzt.
Es gilt unter anderem für diese Legendre-Polynome, dass

$$p_m(1) = 1 \tag{102}$$

ist [54]. Eine weitere Eigenschaft der Legendre-Polynome auf dem Intervall $[-1;1]$ ist durch

$$p_m(-x) = (-1)^m p_m(x) \tag{103}$$

gegeben [54]. Damit erhält man durch Einsetzen des Wertes von $p_m(x)$ an der Stelle $x = 1$, dass ferner

$$p_m(-1) = (-1)^m \tag{104}$$

für alle Legendre-Polynome auf dem Intervall $[-1;1]$ gilt. Mit dieser Erkenntnis und der Rekursionsgleichung (95) kann im Falle der Verwendung von Gauß-Lobatto-Quadratur-Knoten das letzte Element der unteren Nebendiagonale auf Eins gesetzt werden. Betrachtet man die Tatsache, dass $p_m(1) = 1$ ist und dass $p_m(-1) = (-1)^m$ ist, so erhält man mit Hilfe der Drei-Term-Rekursionsformel (88) unter der Annahme, dass die Knoten aus p_m die Quadratur-Knoten sind, einmal für $x = -1$ die Gleichung

$$p_m(-1) = a_m \cdot (-1) \cdot (-1)^{m-1} - c_m \cdot (-1)^{m-2}. \tag{105}$$

Klammert man nun den Term $(-1)^m$ aus, so erhält man

$$p_m(-1) = (-1)^m \cdot (a_m \cdot (-1) \cdot (-1)^{-1} - c_m \cdot (-1)^{-2}). \tag{106}$$

Durch das Ausrechnen einiger Terme ergibt sich schließlich

$$p_m(-1) = (-1)^m \cdot (a_m - c_m). \tag{107}$$

Nun ist allerdings bei der Gauß-Lobatto-Quadratur vorausgesetzt, dass die Punkte $x = -1$ und $x = 1$ fest vorgegebene Quadratur-Knoten sind. Diese Knoten erhält man durch das Berechnen der Nullstellen des Orthogonalpolynoms p_m. Aus diesem Grund ist die Gleichung (107) gleich Null zu setzen, um die gesuchten Nullstellen bzw. Quadratur-Knoten zu erhalten. Daher ergibt sich

$$p_m(-1) = (-1)^m \cdot (a_m - c_m) = 0, \tag{108}$$

was die Erkenntnis liefert, dass

$$a_m = c_m \tag{109}$$

sein muss. Für $x = 1$ kommt man schließlich auf die Gleichung

$$0 = p_m(1) = a_m \cdot 1 \cdot 1 - c_m \cdot 1. \tag{110}$$

Auch hier ist diese angegebene Gleichung äquivalent zu

$$a_m = c_m. \tag{111}$$

Damit ist das Ersetzen des letzten Elements der unteren Nebendiagonale durch 1 zu rechtfertigen.[39] Auf diese Weise erreicht man bei der Berechnung der Quadratur-Knoten bzw. der Nullstellen des Orthogonalpolynoms mittels Lösen des entsprechenden Eigenwertproblems automatisch, dass sich die Rand-Knoten zu 1 und -1 ergeben.

Beispielhaft für $1 \leq m \leq 5$ erhält man nun die Koeffizienten der Tabelle 1.

m	a_m	c_m	b_m
1	1	0	0
2	1.5	0.5	0
3	1.6666	0.6666	0
4	1.75	0.75	0
5	1.8	0.8	0

Tabelle 1: Koeffizienten zum Aufstellen der Matrix T

Mit diesen Angaben ist nun die Matrix T_{GL} der Gauß-Lobatto-Quadratur mit

$$T_{GL} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0.3333 & 0 & 0.6667 & 0 & 0 \\ 0 & 0.4 & 0 & 0.6 & 0 \\ 0 & 0 & 0.4286 & 0 & 0.5714 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix} \tag{112}$$

aufzustellen, wobei hier direkt das letzte Element der unteren Nebendiagonale durch die Zahl 1 ersetzt ist. Ansonsten gibt es keinen Unterschied zwischen der Matrix T_{GL} und der Matrix T_{Leg}.
Nun sind die Eigenwerte von dieser Matrix T_{GL} zu bestimmen. In diesem konkreten Beispiel entsprechen die Eigenwerte den der Größe nach aufsteigend sortierten Knoten, die durch die Werte $-1, -0.6547, 0, 0.6547, 1$ auf dem Intervall $[-1;1]$ gegeben sind.

Bei SDC wird jedoch nicht das Intervall $[-1;1]$ betrachtet. Aus diesem Grund sind die Quadratur-Knoten auf den Bereich $[0;1]$ zu transformieren. Die Transformation eines Knotens auf einen anderen Bereich ergibt sich im Allgemeinen durch

$$Knoten' = Knoten \cdot \frac{b-a}{2} + \frac{b+a}{2}, \tag{113}$$

wobei bei SDC $b = 1$ und $a = 0$ ist. Damit ergeben sich dann im obigen Beispiel die für SDC relevanten Quadratur-Knoten $0, 0.1727, 0.5, 0.8273, 1$. Es ist zu beachten, dass für diese Berechnung der Gauß-Lobatto-Quadratur die Eigenvektoren nicht zur Berechnung der Gewichte

[39] $\frac{c_M}{a_M} = 1$ sofern c_M und a_M gleich sind.

benutzt werden können, da dies nur bei einer Verwendung von orthonormalen Polynomen p_M erlaubt ist. Dies muss beim Gebrauch von orthogonalen Legendre-Polynomen auf eine andere Art und Weise realisiert werden. Siehe dazu [55].

Die Gewichte werden nun durch die Integration der Lagrange-Interpolationspolynome – wie in Gleichung (44) und Gleichung (45) zu sehen ist – berechnet. Jedoch erfolgt bei SDC die Auswertung des Integrals aus Gleichung (44) nicht über Unterintervalle des Zeitraums von 0 bis 1.
Wie in Abbildung 4 zu sehen ist, findet die Auswertung der Funktionen innerhalb eines Intervalls $[t_n;t_{n+1}]$ jeweils über die innerhalb dieses Intervalls liegenden Quadratur-Knoten, die die Unterintervalle $[\tau_m;\tau_{m+1}]$ bilden, statt. Es ist also mit σ_m^l eine Matrix von Gewichten zu berechnen, wobei als τ_m die zuvor berechneten Gauß-Lobatto-Quadratur-Knoten verwendet werden.
Für das Beispiel von fünf Quadratur-Knoten müssen Gewichte für die Intervalle $[0;0.1727]$, $[0.1727;0.5]$, $[0.5;0.8273]$ und $[0.8273;1]$ bestimmt werden. Eine Zeile der Matrix σ_m^l entspricht den Gewichten für eines der genannten Intervalle. Die Dimension der Matrix σ_m^l ist damit 4×5.[40] Allgemein hat sie somit für M Legendre-Gauß-Lobatto-Quadratur-Knoten die Dimension $(M-1)\times M$. Mit dieser Matrix an Gewichten wird in SDC in jedem Korrekturschritt die ausgewertete Rechte-Seite-Funktion multipliziert, um $I_m^{m+1}(\phi^k)$ zu erhalten. $I_m^{m+1}(\phi^k)$ wiederum wird in der finalen diskreten Gleichung (38) zur Implementierung des SDC-Verfahrens, das heißt zum Diskretisieren des Integrals des Fehlermaßes, verwendet.

2.5. Zusammenfassung

In diesem Kapitel wurden die theoretischen Grundlagen zur Implementierung und Verwendung von Spectral Deferred Corrections dargestellt.
Zunächst wurde SDC im Kontinuierlichen analytisch hergeleitet, was zu einer zentralen Korrekturgleichung führte, die es schließlich zu implementieren galt. Um diese jedoch als Basis des numerischen Lösungsverfahrens implementieren zu können, musste SDC diskretisiert werden.
Die Diskretisierung des SDC-Verfahrens umfasste unter anderem die Diskretisierung eines Integrals, welches in der kontinuierlichen Korrekturgleichung enthalten ist. Da dieses Integral nicht geschlossen analytisch berechnet werden konnte und sollte, wurde schließlich die Funktion f durch ein Interpolationspolynom durch gegebene Stützstellen – einem Lagrange-Polynom – angenähert. Mit diesen Lagrange-Polynomen konnten im Anschluss daran Gewichte berechnet werden, die zur numerischen Quadratur benötigt werden. Das Produkt aus diesen Gewichten und der eigentlich zu integrierenden Funktion – ausgewertet an den Stützstellen – ergab die Näherung des ursprünglich zu lösenden Integrals.
Die Punkte, durch die per Interpolation Lagrange-Polynome gelegt wurden, sollten möglichst sinnvoll gewählt werden. Somit fiel die Wahl der Stützstellen auf Punkte mit spektralem Abstand. Konstruiert werden können diese optimalen Punkte durch Orthogonalpolynome. Eine

[40] Allgemeine Angabe der Matrix-Dimension durch $m\times n$, wobei m für die Anzahl der Zeilen und n für die Anzahl der Spalten der Matrix steht.

spezielle Klasse von Orthogonalpolynomen ist die Klasse der Legendre-Polynome, die in dieser Arbeit gewählt wurden. Aus den Nullstellen der Legendre-Polynome resultierten die spektralen Knoten und die zugehörigen Gewichte können entsprechend berechnet werden. Dieses Verfahren wird auch als Gauß-Legendre-Quadratur bezeichnet. Zusätzlich war es bei SDC wünschenswert, einen Startpunkt und einen Endpunkt auf einem Intervall zu fixieren, so wurde die Legendre-Gauß-Lobatto-Quadratur gewählt, eine kleine Abwandlung der klassischen Gauß-Legendre-Quadratur. Mit Hilfe dieser Form der Integral-Diskretisierung, die auf effiziente Weise mit Hilfe von Eigenwerten durchgeführt werden kann, wurde dann das komplette SDC-Verfahren diskretisiert und im Anschluss daran implementiert. Abschließend ist die iterative Vorgehensweise von SDC per Pseudocode dargestellt worden.

Es wurde also in diesem Kapitel das numerische Lösungsverfahren Spectral Deferred Corrections vorgestellt und hergeleitet. Numerische Lösungsverfahren sollen immer möglichst nah an der exakten Lösung des Problems liegen, damit sie sinnvoll als Näherungsverfahren eingesetzt werden können. Bei einer adäquaten numerischen Lösungsmethode muss folglich immer Konvergenz gegen die exakte Lösung gegeben sein. Ansonsten ermittelt man zwar eine Lösung mit dem Algorithmus des numerischen Verfahrens, welche jedoch mit der eigentlichen Lösung des Problems nicht im Zusammenhang steht.
Zu prüfen ist demzufolge, ob SDC ein konvergentes und zudem stabiles numerisches Lösungsverfahren ist, so dass es taugliche Ergebnisse liefert. Infolgedessen wird im nachfolgenden Kapitel unter anderem auf die Konvergenz, auf die damit verbundenen Konvergenzordnungen, auf die Genauigkeit der numerischen Lösung per SDC u. ä. eingegangen.

3. Konvergenz- und Stabilitätsbetrachtungen für das allgemeine Testproblem

In diesem Kapitel wird zunächst der Begriff der Konvergenz erörtert. Nachfolgend wird SDC auf Konvergenz hin untersucht. Der zweite Analyseaspekt innerhalb dieses Kapitels konzentriert sich auf die Stabilität und die Stabilitätsgebiete von SDC.
Bis zu diesem Zeitpunkt war in der vorliegenden Arbeit nur von einer genutzten Rechteckregel die Rede. Auf diese Basis wird zu Beginn dieses Kapitels genauer Bezug genommen.

3.1. Konvergenzanalyse

Im Allgemeinen beschreibt der Begriff 'Konvergenz' den Zusammenhang zwischen einer exakten Lösung und durch ein numerisches Verfahren angenäherten Lösung.

Nach Dutt ist die Definition von Konvergenz, die direkt auf das SDC-Verfahren zugeschnitten ist, durch

$$\|\widetilde{\phi}(b) - \phi(b)\| < C \cdot (b-a)^{p+1} \tag{114}$$

gegeben.[41] Dabei wird das Anfangswertproblem auf dem Intervall $[a;b]$ einmal approximativ mit einem numerischen Verfahren $\widetilde{\phi}$, welches hier SDC ist, ausgewertet. Und es wird das Anfangswertproblem über den gleichen Zeitbereich durch ϕ exakt gelöst. Die Auswertung zum Urteilen über die Konvergenzordnung findet an dem Intervallendpunkt statt. Existiert eine Konstante $C > 0$, so dass die Gleichung (114) erfüllt ist, dann hat das numerische Verfahren bei ausreichend glatter Rechte-Seite-Funktion die Konvergenzordnung p [31].

Im Unterkapitel 2.3.3 wurde bereits erwähnt und begründet, dass bei der Gauß-Lobatto-Quadratur die Konvergenzordnung $p = 2M - 2$ beträgt. Es wird angenommen, dass auch SDC mit verwendeten Gauß-Lobatto-Quadratur-Knoten für M Quadratur-Knoten mit K Iterationen die Ordnung $min(K, 2M-2)$ aufweist.
In diesem Kapitel aufgeführte Konvergenzanalysen werden den Konvergenzbereich von SDC untersuchen und die Annahme der Konvergenzordnung entsprechend der Anzahl an Quadratur-Knoten im Fall von Konvergenz bestätigen. Die Tatsache, dass für das Konvergenzverhalten von SDC mit Gauß-Lobatto-Quadratur $p = 2M - 2$ gilt, ist eine wichtige Erkenntnis, die als Eingabe in das SDC-Programm zu späteren Analysen genutzt wird. In Kapitel 4 wird das SDC-Verfahren unter verschiedenen Anzahlen von Quadratur-Knoten M so lange iteriert, bis es konvergiert und nahezu die maximal mögliche Konvergenzordnung erreicht hat. Es ergibt sich also mit der bekannten Konvergenzordnung p für die Anzahl der Iterationen eines SDC-Durchlaufs die Gleichung $K = 2M - 2$.

Die Spectral Deferred Corrections Methode kann mit einfachen Basis-Verfahren auf drei verschiedene Weisen implementiert und genutzt werden. Bisher wurde zur Herleitung von SDC

[41] Dutt gibt in seiner Veröffentlichung [31] keine spezielle Norm bezüglich der Gleichung (114) vor. Daher kann hier eine beliebige Norm gewählt werden.

die rechtsseitige Rechteckregel als Basis-Verfahren genutzt. Weitere einfache Basis-Verfahren in der Numerik sind die linksseitige Rechteckregel und eine Kombination aus linksseitiger und rechtsseitiger Rechteckregel. Betrachtet man diese Ansätze genauer, so stellen diese Rechteckregeln zur Approximation eines Integrals bekannte numerische Standard-Verfahren dar. Diese sind bekannt als

1. explizites Euler-Verfahren
2. implizites Euler-Verfahren
3. semi-implizites Euler-Verfahren

und stehen im Folgenden für die Art der verwendeten Rechteckregel zur Diskretisierung des Integrals innerhalb des SDC-Verfahrens. Bei der Verwendung dieser Verfahren in SDC sind jeweils nur geringfügige Änderungen im Programm durchzuführen, welche große Auswirkungen auf das Ergebnis haben. Nachstehend wird SDC auf die Testgleichung aus Unterkapitel 1.2 mit $\lambda = -1$ angewandt. Falls nicht anders erwähnt, wird diese auf dem Zeitraum $t \in [0;1]$ untersucht.[42]

3.1.1. Explizites Euler-Verfahren

Das explizite Euler-Verfahren hat im Vergleich zu den beiden anderen genannten den geringsten Rechenaufwand, da die Lösung – wie der Name schon sagt – explizit berechnet werden kann und keine impliziten Abhängigkeiten in der Gleichung vorkommen. Das Integral wird dabei durch die linksseitige Rechteckregel relativ grob angenähert und man erhält

$$\phi(t+\Delta t, t, u) = u(t) + \Delta t \cdot f(t, u(t)) \tag{115}$$

als numerische Evolution [19].[43] Bei iterativer Durchführung nutzt man

$$u_{i+1} = u_i + h \cdot f(t, u_i) \tag{116}$$

zur Berechnung der Lösung [19].
Der entscheidende Nachteil des expliziten Euler-Verfahrens ist jedoch das geringe Stabilitätsgebiet und die Nicht-Tauglichkeit für steife Differentialgleichungen [19]. Im Stabilitätsbereich, bei ausreichend kleiner Schrittweite Δt, ist die Konvergenzordnung bzw. die Konvergenzgeschwindigkeit identisch mit SDC bei zugrunde liegendem impliziten Euler-Verfahren.[44] Bei Instabilität des expliziten Euler-Verfahrens, welche relativ schnell durch eine zu große Schrittweite Δt erreicht wird [19], ist der Fehler in der relativen Maximumsnorm sehr groß.[45] In Abbildung 5 kann beobachtet werden, dass bei einer kleinen Anzahl an Zeitschritten N, in die

[42] Da SDC in Unterkapitel 3.1 auf die Testgleichung mit $\lambda = -1$ angewendet wird und dadurch kein Splitting realisiert werden kann, wird hier das semi-implizite Euler-Verfahren als Basis-Verfahren erst einmal nur erwähnt und nicht weiter analysiert. Die Untersuchung dieser kombinierten Euler-Variante erfolgt im weiteren Verlauf der Arbeit anhand der SWFW-Gleichung.

[43] Die Variable Δt steht auch hier für die Schrittweite mit der t erhöht wird.

[44] Daher wird im Unterkapitel 3.1.2 eine gemeinsame Betrachtung der Konvergenz von implizitem und explizitem Euler-Verfahren durchgeführt.

[45] $err = \frac{max\|u-exakt\|_\infty}{max\|exakt\|_\infty}$

das zu betrachtende Intervall unterteilt wird, SDC mit explizitem Euler-Verfahren nicht konvergiert. Zu sehen ist ein sehr großer Fehler in relativer Maximumsnorm. Ab einer ausreichend kleinen Schrittweite Δt bzw. einer genügend großen Anzahl N von Zeitschritten konvergiert das SDC-Verfahren auf Basis des expliziten Euler-Verfahrens. Auch dies wird in Abbildung 5 ersichtlich.

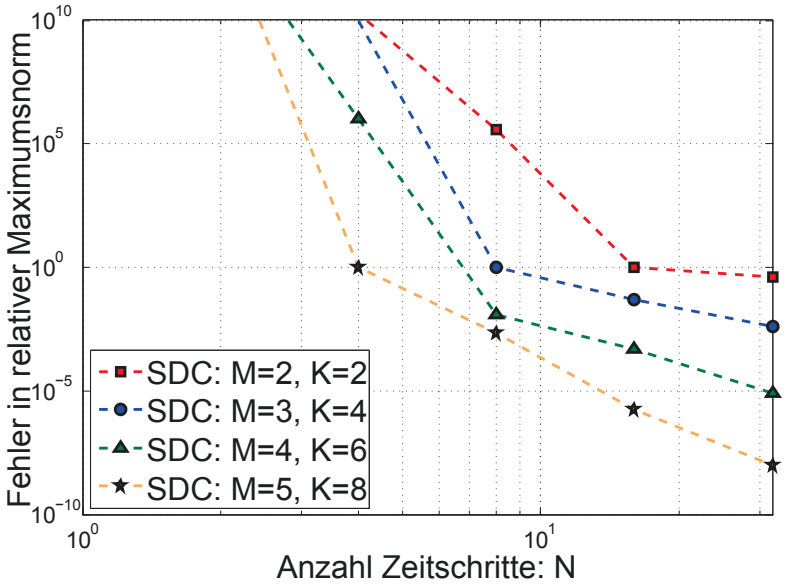


Abbildung 5: SDC auf Basis des expliziten Euler-Verfahrens bei zunächst zu großer Schrittweite Δt mit Sichtbarkeit des Übergangs von nicht konvergentem zu konvergentem Verhalten

In Abbildung 5 wächst nicht nur die Anzahl der Quadratur-Knoten M, sondern wie auf der x-Achse zu erkennen ist, auch die Anzahl an Zeitschritten N. In diesem Beispiel wird der Gesamtzeitraum von 0 bis 15 untersucht. Bei wenigen Unterteilungen des Intervalls $[0;15]$ ist die Schrittweite bei SDC auf Basis des expliziten Euler-Verfahrens zu groß und das Verfahren ist instabil [19]. Zu Beginn liegt daher ein großer Fehler in relativer Maximumsnorm vor. Je kleiner jedoch die einzelnen betrachteten Intervalle werden – je größer N wird – desto besser konvergiert das Verfahren.[46]
Wie das Stabilitätsgebiet für das explizite Euler-Verfahren berechnet werden kann und wie es in der konkreten SDC-Approximation aussieht, wird in Kapitel 3.2 genauer erklärt.

3.1.2. Implizites Euler-Verfahren

Beim impliziten Euler-Verfahren haben wir keine Einschränkungen an die Schrittweite, um Stabilität zu gewährleisten [19]. Ein großer Nachteil ist jedoch der hohe Rechenaufwand dieses Verfahrens, da – wie der Name schon sagt – implizite Abhängigkeiten bestehen. Die Approximation des Integrals geschieht bei der impliziten Euler-Variante per rechtsseitiger Recht-

[46] Es konvergiert erst ab einer hinreichend kleinen Intervalllänge überhaupt.

eckregel, so dass man

$$u_{i+1} = u_i + \Delta t\, f(t, u_{i+1}) \tag{117}$$

als Auswertungsvorschrift für die numerische Evolution erhält. Es ist sofort ersichtlich, dass eine implizite Auswertung erfolgen muss, da u_{i+1} auf der linken Seite der Gleichung noch einmal in der Funktionsauswertung der rechten Seite vorkommt. Daher muss in jedem Schritt eine im Allgemeinen nichtlineare Gleichung gelöst werden,[47] was hohen Rechenaufwand zur Folge hat [19].

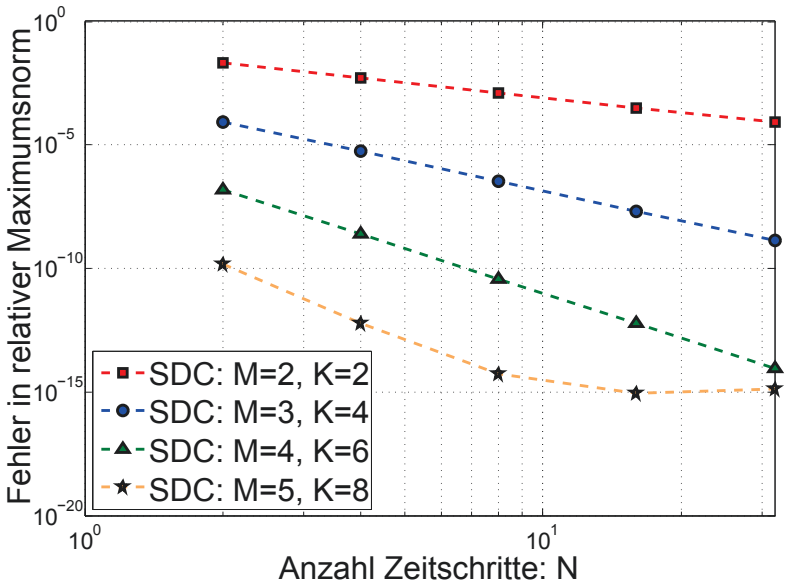


Abbildung 6: SDC auf Basis des impliziten Euler-Verfahrens bzw. expliziten Euler-Verfahrens bei immer existierender ausreichend kleiner Schrittweite Δt. Dabei sind die durch die variierende Anzahl an Quadratur-Knoten M entsprechenden Konvergenzordnungen zu erkennen.

Abbildung 6 zeigt das Konvergenzverhalten bei der Anwendung des impliziten Euler-Verfahrens.[48] Je mehr Teilintervalle N gewählt werden, desto kleiner ist der Fehler, auch schon bei der ersten Iteration. Zudem wird ersichtlich, dass die gestrichelten Geraden für die Fehler mit wachsendem M mit gleichzeitig wachsendem K immer steiler werden. Das bedeutet, dass die numerische Lösung bei SDC mit höherer Anzahl an Quadratur-Knoten und gleichzeitig mehr Korrekturschritten schneller konvergiert und demzufolge eine höhere Konvergenzordnung besitzt. Die unterste Gerade für $M = 5$ und $K = 8$ verfälscht ein wenig das nachstehende Ergebnis für die Konvergenzordnung, da bei einem Fehler von circa 10^{-15} bis 10^{-16} bereits die Maschinengenauigkeit erreicht ist.

[47] Ein lineares Gleichungssystem erhält man nur im Sonderfall, wenn die Funktion f linear ist.

[48] In diesem Fall ist die Konvergenzordnung von SDC mit explizitem Euler-Verfahren identisch mit der Konvergenzordnung von SDC mit implizitem Euler-Verfahren.

Nach der Betrachtung dieser Abbildung, kann die theoretische Annahme für die Konvergenzordnung p für die Gauß-Lobatto-Quadratur aus dem Unterkapitel 2.2 in der Praxis bestätigt werden. Dazu wird die Konvergenzordnung innerhalb des Matlab-Programms mit Hilfe der Formel (17) berechnet. Es ergeben sich auf diese Weise die Werte aus Tabelle 2 durch Mittelung der berechneten Konvergenzordnungen auf den vier Abschnitten zwischen den eingezeichneten Markern.[49]

Quadratur-Knoten: M	2	3	4	5
Konvergenzordnung (Theorie): p	2	4	6	8
Konvergenzordnung (Realität): p	2.2334	4.0084	6.0842	7.9275

Tabelle 2: Konkrete Konvergenzbetrachtung für das implementierte SDC-Verfahren im Vergleich zu den theoretischen Annahmen

3.2. Stabilitätsanalyse

Allgemein gilt es mit numerischen Verfahren die Lösung der exakten Evolution

$$\phi(h,t,u) = \phi(t+h,t,u) = u(t) + \int_t^{t+h} f(\tau, u(\tau))d\tau \tag{118}$$

mit möglichst kleinem Rechenaufwand möglichst genau zu berechnen [19].

Dutt definiert Stabilität über einen sogenannten 'Amplifikationsfaktor' $Am(\lambda)$.[50] Dieser Amplifikationsfaktor für $\lambda \in \mathbb{C}$ ist für eine Auswertung auf dem Intervall $[0;1]$ und $\Delta t = 1$ durch

$$Am(\lambda) = \frac{\widetilde{\phi}(1)}{\widetilde{\phi}(0)} = \widetilde{\phi}(1) \tag{119}$$

definiert, wobei sich die letzte Gleichheit durch die Tatsache ergibt, dass der Anfangswert $\widetilde{\phi}(0) = e^{\lambda \cdot 0}$ dem Wert 1 entspricht. Wenn also ein λ durch die Problemstellung gegeben ist, heißt ein numerisches Verfahren nach Dutt stabil, wenn

$$|\, Am(\lambda)\,| = |\, \widetilde{\phi}(1)\,| \leq 1 \tag{120}$$

ist [31]. Direkt aus dieser Definition resultieren zwei weitere Stabilitätsbegriffe. Ein Verfahren ist A-stabil, wenn es für alle λ aus der linken komplexen Halbebene stabil ist [31] und ein Verfahren heißt L-stabil, falls

$$\lim_{Re(\lambda)\to-\infty} Am(\lambda) = 0 \tag{121}$$

[49] Die Berechnung der Konvergenzordnung p bei $M = 5$ und $K = 8$ erfolgt durch die Mittelung der Konvergenzordnungen zwischen den ersten drei Markern, da der restliche Kurvenverlauf durch Erreichen der Maschinengenauigkeit die Konvergenzordnung verfälscht.

[50] Das λ bezüglich dem der Amplifikationsfaktor berechnet wird, entspricht dem λ aus der Differentialgleichung $\phi'(t) = \lambda \cdot \phi(t)$ des gegebenen Anfangswertproblems.

gilt [31]. Neben dem starken Begriff der A-Stabilität existiert eine weitere, ähnliche Definition. Man nennt ein numerisches Verfahren A(α)-stabil, sofern es für alle λ stabil ist und $\pi - \alpha \leq arg(\lambda) \leq \pi + \alpha$ gilt. Der etwas abgeschwächte Begriff der A(α)-Stabilität entspricht genau dann der A-Stabilität, wenn der Winkel $\alpha = 90° = \frac{\pi}{2}$ beträgt [31].[51]

Hinsichtlich des Qualitätsmerkmals der Stabilität wird SDC auf Basis der drei Euler-Varianten untersucht.

3.2.1. Explizites Euler-Verfahren

Bei den folgenden Stabilitätsbetrachtungen für SDC auf Basis des expliziten Euler-Verfahrens ist die Schrittweite Δt fest vorgegeben und der interessierende Parameter ist λ. Grundsätzlich gilt die Testgleichung für $\lambda \in \mathbb{C}$. Interessiert ist man nun daran solche λ zu ermitteln, für die SDC mit explizitem Euler-Verfahren stabil ist.

Dies wurde zunächst für SDC mit explizitem Euler als Basis-Verfahren und $M = 3$, $K = 4$ und $N = 1$ ermittelt. Die Ergebnisse sind in Abbildung 7 in Form des Stabilitätsgebiets und zwei Genauigkeitsgebieten zu sehen. Dabei beschränken sich alle folgenden Abbildungen auf die positive imaginäre Achse, da sowohl die Stabilitäts- als auch die Genauigkeitsgebiete symmetrisch um die reelle Achse liegen.

Für alle λ innerhalb der äußeren geschlossenen Form in durchgezogener, roter Linie in Abbildung 7 ist das Verfahren stabil. Sofort ersichtlich ist, dass das Verfahren nicht A-stabil ist, da unendlich viele λ außerhalb dieser Fläche, aber in der negativen komplexen Halbebene liegen. Zusätzlich ist in Abbildung 7 die Genauigkeit der Approximation innerhalb des Intervalls $[0;1]$ zu erkennen. Diese berechnet sich aus $\mid \widetilde{\phi}(1) - \phi(1) \mid \leq \varepsilon$.
Es gilt von der Näherungslösung per SDC an der rechten Intervallgrenze die exakte Lösung zu gleicher Zeit t zu subtrahieren [31]. Die Bereiche, in denen der Betrag dieser Abweichung kleiner ist als ein gegebenes $\varepsilon > 0$, sind für zwei konkrete ε mit $\varepsilon = 0.01$ und $\varepsilon = 0.0001$ ebenfalls eingezeichnet.

Auffällig ist die sehr schnell schrumpfende Genauigkeit. Je größer der Realteil oder der Imaginärteil von λ wird, desto weniger bzw. gar nicht konvergiert das Verfahren. Möchte man z. B. per SDC mit explizitem Euler-Verfahren und $M = 3$, $K = 4$ und $N = 1$ eine Lösung der Testgleichung erhalten, die einen kleineren Unterschied als $\varepsilon = 0.01$ von der exakten Lösung hat, so darf λ bei nicht existierendem Imaginärteil im Intervall $[-1.5;1]$ liegen. Wird der Imaginärteil größer oder kleiner als Null, so wird das Intervall für den zugehörigen Realteil von λ immer kleiner. Die Schrittweite ist in diesem Fall zu groß, um eine ausreichende Genauigkeit der Approximation zu erhalten.

[51] Diese Definition der A(α)-Stabilität ist gleichbedeutend damit, dass ein Verfahren A(α)-stabil ist, wenn sein Stabilitätsgebiet einen symmetrischen Kegel um die negative reelle Achse mit einem Scheitelwinkel von 2α enthält [19].

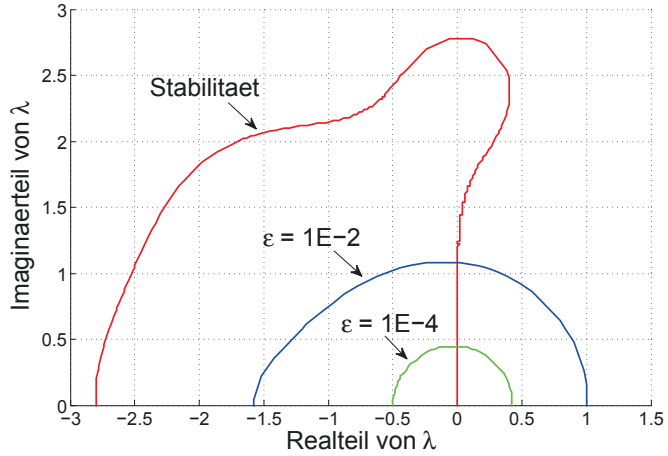


Abbildung 7: Stabilitäts- und Genauigkeitsgebiete bei SDC mit explizitem Euler-Verfahren und $M = 3$, $K = 4$, $N = 1$ zu $\varepsilon = 10^{-2}$ und $\varepsilon = 10^{-4}$

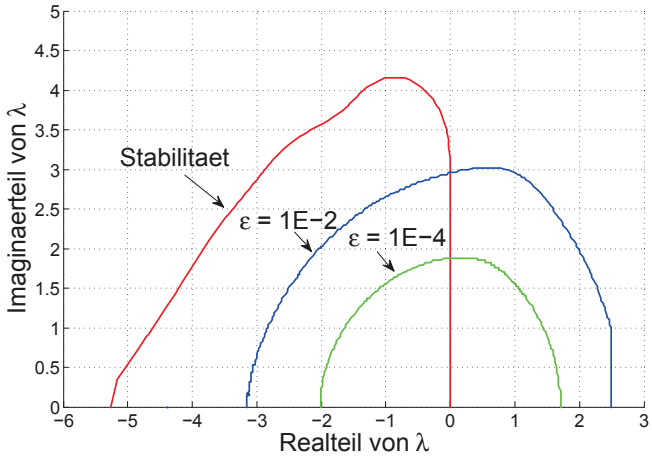


Abbildung 8: Stabilitäts- und Genauigkeitsgebiete bei SDC mit explizitem Euler-Verfahren und $M = 5$, $K = 8$, $N = 1$ zu $\varepsilon = 10^{-2}$ und $\varepsilon = 10^{-4}$

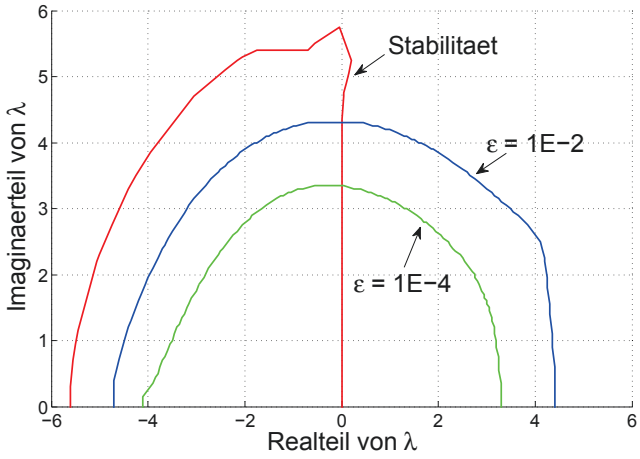


Abbildung 9: Stabilitäts- und Genauigkeitsgebiete bei SDC mit explizitem Euler-Verfahren und $M = 7$, $K = 12$, $N = 1$ zu $\varepsilon = 10^{-2}$ und $\varepsilon = 10^{-4}$

Im nächsten Schritt erfolgen erneute Auswertungen von SDC mit dem explizitem Euler-Verfahren als Basis-Verfahren. Jedoch wurden die Parameter M und K variiert. Die zugehörigen Parameter zu Abbildung 8 lauten $M = 5$, $K = 8$ und $N = 1$. Durch die höhere Anzahl an Quadratur-Knoten und an Iterationen vergrößert sich das Stabilitätsgebiet und die Genauigkeit der numerischen Ergebnisse verbessert sich. Hat man zuvor bei $M = 3$ und $K = 4$ ein Stabilitätsgebiet für λ auf der reellen Achse bis circa -3.8 und auf der imaginären Achse von circa -2.75 bis 2.75 an der maximalen Stelle, so wächst dieses Gebiet für $M = 5$ und $K = 8$ für den Realteil auf circa -5.3 bis 0 und den Imaginärteil maximal auf circa -4.25 bis 4.25. Zum besseren Vergleich der beiden SDC-Varianten sind die Epsilonwerte in Abbildung 8 identisch mit denen in Abbildung 7. Bei SDC mit $M = 5$ und $K = 8$ sind diese Genauigkeitsgebiete bereits wesentlich größer als die Genauigkeitsgebiet bei $\varepsilon = 10^{-2}$ und $\varepsilon = 10^{-4}$ für SDC mit $M = 3$ und $K = 4$. Ihre Größe hat sich in beiden Fällen mehr als verdoppelt.

In Abbildung 9 sind abschließend das Stabilitätsgebiet und die Genauigkeitsgebiete für SDC mit $M = 7$, $K = 12$ und $N = 1$ basierend auf dem expliziten Euler-Verfahren zu sehen. Beim Stabilitätsgebiet wächst vor allem der zulässige Imaginärteil von λ noch einmal zusätzlich an. Ein großer Imaginärteil, der Probleme hinsichtlich der Stabilität verursachen könnte, kann durch eine größere Anzahl von Quadratur-Knoten M und Korrekturschritten K kompensiert werden. Auch die Flächen innerhalb der Genauigkeitsgebiete haben sich im Vergleich zu Abbildung 8 erneut fast verdoppelt.

Zusammenfassend ist jedoch zu sagen, dass das Stabilitätsgebiet bei einer zu großen Schrittweite Δt und geringer Anzahl an Quadratur-Knoten M ziemlich klein ist. Weiterhin ist das Verfahren ebenfalls nicht A-stabil.[52]

[52] wie alle expliziten Euler-Verfahren

3.2.2. Implizites Euler-Verfahren

In diesem Unterkapitel wird SDC auf Basis des impliziten Euler-Verfahrens auf die Testgleichung angewendet.

Die drei Abbildungen 10, 11 und 12 zeigen – äquivalent zu den drei Abbildungen aus dem vorangehenden Kapitel – die resultierenden Genauigkeits- und Stabilitätsgebiete für SDC auf Basis des impliziten Euler-Verfahrens.

In Abbildung 10 ist SDC mit $M = 3$, $K = 4$ und $N = 1$ analysiert worden. Es ist zu erkennen, dass die Trennlinie für das Stabilitätsgebiet eine nahezu senkrechte Gerade durch den Nullpunkt ist. Bei größer werdendem Imaginärteil von λ hat sie eine leichte Krümmung nach rechts. Dies spiegelt ein wenig den Verlauf des Stabilitätsgebiets vom reinen impliziten Euler-Verfahren wider. Dieses entspricht einem Kreis mit Mittelpunkt 1 und Radius 1 um die reelle Achse [19]. Diese leichte Krümmung beeinflusst jedoch nicht die Tatsache, dass das Verfahren für alle λ mit Realteil kleiner oder gleich Null stabil ist.[53] Das ist von großer Wichtigkeit, da dies die Bedingung für A-Stabilität ist. SDC auf Basis des impliziten Euler-Verfahrens ist – genauso wie das 'einfache' implizite Euler-Verfahren – A-stabil. Dies ist ein gravierender und bedeutender Unterschied zu SDC mit explizitem Euler-Verfahren.
Zusätzlich sind auch hier, wie bei den Auswertungen bezüglich des expliziten Euler-Verfahrens, Genauigkeitsgebiete zu $\varepsilon = 0.01$ und $\varepsilon = 0.0001$ eingezeichnet. Diese weisen jedoch im Vergleich zum expliziten Euler-Verfahren mit $M = 3$, $K = 4$ und $N = 1$ keine nennenswerten Veränderungen bzw. Verbesserungen auf. Der entscheidende Unterschied – um das hier noch einmal herauszustellen – ist die Stabilitätseigenschaft des impliziten Euler-Verfahrens.

Die Abbildung 11 zeigt die gleiche Stabilitätseigenschaft für SDC basierend auf dem impliziten Euler-Verfahren mit $M = 5$ und $K = 8$ wie zuvor für SDC mit $M = 3$ und $K = 4$. Auffallend ist jedoch die Vergrößerung der Genauigkeitsgebiete. Die Ellipsen, die hier aus Symmetriegründen nur für die positiven Imaginärwerte aufgetragen sind, haben sich für die Genauigkeitsbedingung zu $\varepsilon = 0.01$ von einer maximalen Ausdehnung auf der reellen Achse von circa $[-2.7; 0.6]$ und einer maximalen Ausdehnung auf der imaginären Achse von circa $[-2; 2]$ für $M = 3$ und $K = 4$ auf circa $[-8.2; 1.2]$ auf der reellen Achse und circa $[-5.4; 5.4]$ auf der imaginären Achse für $M = 5$ und $K = 8$ vergrößert. Der Umfang entspricht also bei zwei zusätzlichen Quadratur-Knoten und der Verdopplung der Korrekturschritte fast dem Dreifachen.[54]

Zur Vollständigkeit zeigt die Abbildung 12 das Stabilitätsgebiet und die Genauigkeitsgebiete für SDC basierend auf dem impliziten Euler-Verfahren mit $M = 7$, $K = 12$ und $N = 1$. Das Stabilitätsgebiet verändert sich nicht. Die Genauigkeitsgebiete wachsen erneut enorm an und können von der Abbildung aufgrund ihrer Größe nur noch teilweise erfasst werden.

[53] Alle λ mit $Re(\lambda) > 0$ interessieren hier nicht, da sie für die A-Stabilität keine Rolle spielen.
[54] Aus der Abbildung 10 und der Abbildung 11 ist eine ähnliche Erkenntnis für die Genauigkeitsbedingung zu $\varepsilon = 0.0001$ zu gewinnen.

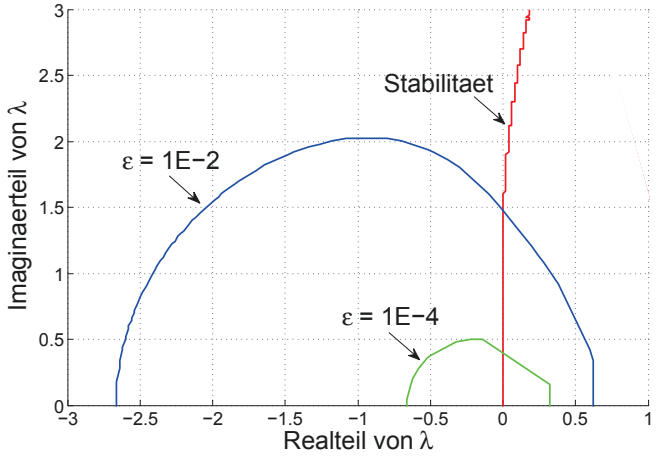


Abbildung 10: Stabilitäts- und Genauigkeitsgebiete bei SDC mit implizitem Euler-Verfahren und $M = 3$, $K = 4$, $N = 1$ zu $\varepsilon = 10^{-2}$ und $\varepsilon = 10^{-4}$

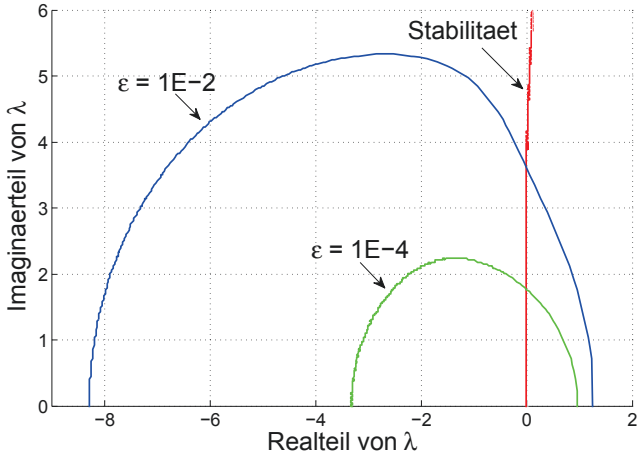


Abbildung 11: Stabilitäts- und Genauigkeitsgebiete bei SDC mit implizitem Euler-Verfahren und $M = 5$, $K = 8$, $N = 1$ zu $\varepsilon = 10^{-2}$ und $\varepsilon = 10^{-4}$

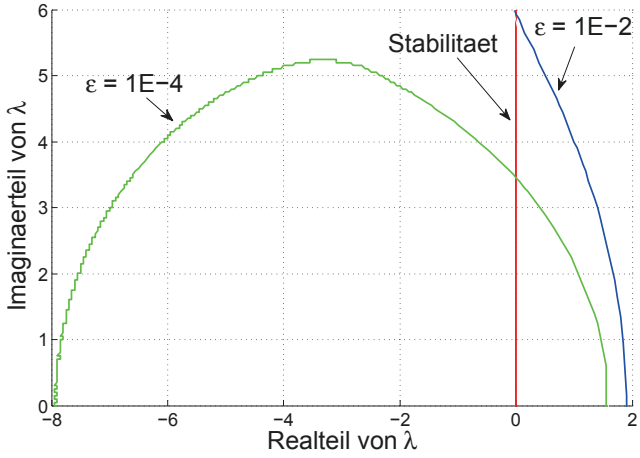


Abbildung 12: Stabilitäts- und Genauigkeitsgebiete bei SDC mit implizitem Euler-Verfahren und $M = 7$, $K = 12$, $N = 1$ zu $\varepsilon = 10^{-2}$ und $\varepsilon = 10^{-4}$

3.2.3. Semi-implizites Euler-Verfahren

In der speziellen semi-impliziten Variante von SDC, welche SDC zu einem besonders wertvollen Verfahren für bestimmte Fragestellungen macht, wird eine Kombination aus den gerade angesprochenen expliziten und impliziten Euler-Verfahren verwendet. Die rechte Seite der Differentialgleichung wird in eine Summe aufgespalten, wobei ein Summand schnell veränderliche Komponenten und ein Summand langsam veränderliche Komponenten beinhaltet. Wie schon erwähnt, taugt das explizite Euler-Verfahren nicht für steife Differentialgleichungen. Es ist jedoch wesentlich günstiger in der Rechenzeit und im Aufwand. Daher versucht man alle Gleichungsterme der Differentialgleichung, die sich zeitlich nicht so schnell ändern, explizit zu berechnen und verwendet nur da wo es unbedingt nötig ist – für die schnell veränderlichen Terme – das aufwendigere implizite Euler-Verfahren.
Insgesamt ergibt sich

$$u_{i+1} = u_i + h \cdot (f_I(t, u_{i+1}) + f_E(t, u_i)) \tag{122}$$

als abstrakte Rechenvorschrift für das semi-implizite Euler-Verfahren. Details zur konkreten Implementierung in SDC wurden in Kapitel 2.2.1 dargestellt.

Auch diese dritte Variante von SDC ist mit der Testgleichung getestet worden und auf Stabilitäts- und Genauigkeitsgebiete untersucht worden. λ ist in eine Summe aus explizitem und implizitem Teil gespalten, also $\lambda = (\alpha + \imath\beta)$ [26]. Dabei wird der Realteil implizit und der Imaginärteil explizit behandelt. Diese Aufteilung wird so gewählt, da die Komponenten zu den Eigenwerten mit relativ betragsgroßen Realteilen die steifen Komponenten darstellen und somit implizit behandelt werden müssen. In Anlehnung an den Artikel von Minion 2003 [26] werden in diesem Unterkapitel 3.2.3 Untersuchungen hinsichtlich des semi-impliziten Stabilitätsgebiets durchgeführt.

In Abbildung 13 wurde SDC mit dem semi-impliziten Euler-Verfahren mit $M = 3$, $K = 4$ und $N = 1$ ausgeführt. In dieser Variante wird teilweise das explizite Euler-Verfahren genutzt, welches nicht A-stabil ist. Teilweise wird das implizite Euler-Verfahren genutzt, welches sehr wohl A-stabil ist. Daraus resultiert die Stabilitätslinie, die senkrecht durch den Nullpunkt der reellen Achse verläuft bis der Imaginärteil betragsmäßig größer als circa 1.5 ist. Danach setzt sich das Stabilitätsverhalten des expliziten Euler-Verfahrens durch. Daher ist das semi-implizite Verfahren nicht A-stabil, sondern nur A(α)-stabil [26]. Bei einem Realteil von -5 liegt der entsprechende Imaginärteil des Stabilitätsgebiets bei circa 5.3.
Die Genauigkeitsgebiete sind eine Kombination aus dem komplett impliziten Euler-Verfahren und dem komplett expliziten Euler-Verfahren. Die Intervalllänge auf der reellen Achse für einen Imaginärteil von 0 entspricht der Intervalllänge des impliziten Euler-Verfahrens. Wohingegen die Intervalllänge auf die maximale Ausdehnung der Genauigkeitsgebiete in Richtung der imaginären Achse identisch mit der des expliziten Euler-Verfahrens ist. Es liegen also bei SDC mit semi-implizitem Euler-Verfahren ebenfalls geschlossene Ellipsen vor, die ein Mittel aus explizitem und implizitem Euler bilden und hier aus Gründen der Symmetrie nur für die positiven Imaginärteile von λ abgebildet sind.

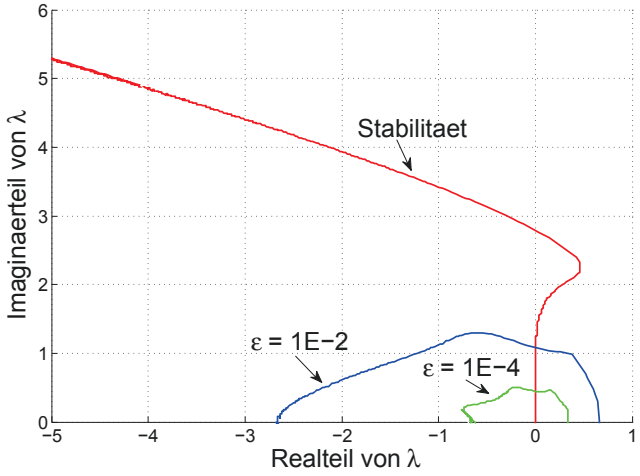


Abbildung 13: Stabilitäts- und Genauigkeitsgebiete bei SDC mit semi-implizitem Euler-Verfahren und $M = 3$, $K = 4$, $N = 1$ zu $\varepsilon = 10^{-2}$ und $\varepsilon = 10^{-4}$

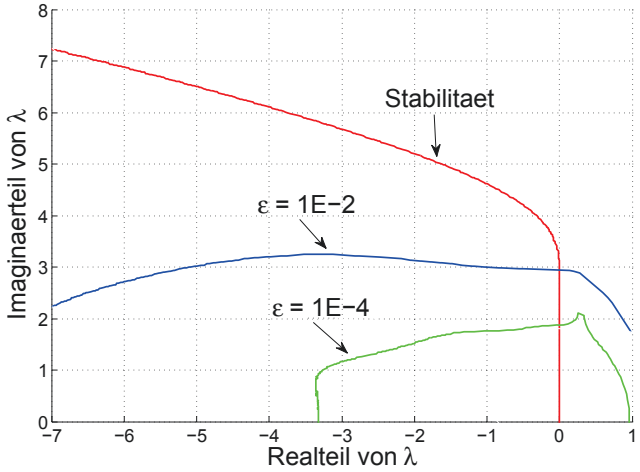


Abbildung 14: Stabilitäts- und Genauigkeitsgebiete bei SDC mit semi-implizitem Euler-Verfahren und $M = 5$, $K = 8$, $N = 1$ zu $\varepsilon = 10^{-2}$ und $\varepsilon = 10^{-4}$

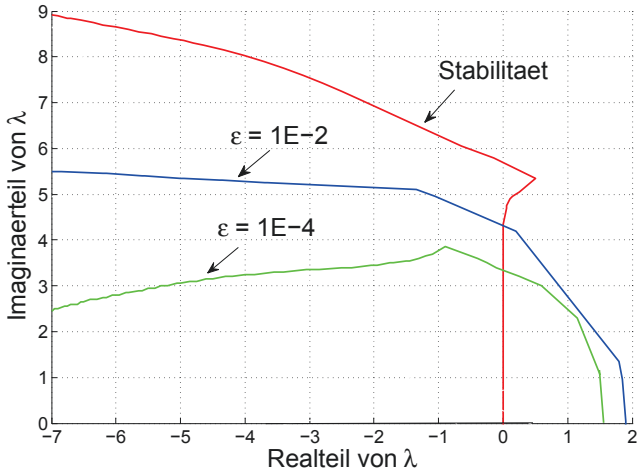


Abbildung 15: Stabilitäts- und Genauigkeitsgebiete bei SDC mit semi-implizitem Euler-Verfahren und $M = 7$, $K = 12$, $N = 1$ zu $\varepsilon = 10^{-2}$ und $\varepsilon = 10^{-4}$

Abbildung 14 zeigt Ergebnisse für SDC basierend auf dem semi-impliziten Euler-Verfahren zu $M = 5$, $K = 8$ und $N = 1$. Es ist zu erkennen, dass das Stabilitätsgebiet größer geworden ist und im Vergleich zur Auswertung mit $M = 3$ und $K = 4$ aus Abbildung 13 bei einem Realteil von λ von -5 bereits ein Imaginärteil von circa 6.5 zulässig ist, um ein stabile Lösung zu gewährleisten. Der Winkel der A(α)-Stabilität hat sich also etwas vergrößert zum vorher beschriebenen Fall mit drei Quadratur-Knoten.
Die Genauigkeitsgebiete wachsen bei steigender Anzahl an Quadratur-Knoten und Iterationsschritten auf der reellen Achse äquivalent zu denen des rein impliziten Euler-Verfahrens und auf der imaginären Achse äquivalent zu denen des rein expliziten Euler-Verfahrens an.

Auch bei SDC mit semi-implizitem Euler-Verfahren ist zur Vollständigkeit die Abbildung 15 für $M = 7$, $K = 12$ und $N = 1$ abgedruckt. Eine Vergrößerung des Stabilitätsgebiets und der Genauigkeitsgebiete durch eine wachsende Anzahl an Quadratur-Knoten M und Korrekturschritten K ist auch hier festzustellen.

3.3. Zusammenfassung

In Anlehnung an die Veröffentlichungen von Dutt, Greengard, Rokhlin [31] und Minion [26] wurden mittels des selbst implementierten SDC-Verfahrens, wie in Kapitel 2 beschrieben, Untersuchungen hinsichtlich der Konvergenz und Stabilität durchgeführt.
Diese Analyse zur Reproduktion der Ergebnisse von Dutt et al. und Minion nutzt die allgemeine Testgleichung. Auf diese Testgleichung wird SDC mit drei verschiedenen Basis-Verfahren angewandt. Zunächst erfolgt die Untersuchung von SDC mit explizitem Euler-Verfahren als Basis-Verfahren, gefolgt vom impliziten Euler-Verfahren. Abschließend wird eine Kombination aus explizitem und implizitem Euler-Verfahren analysiert. Festzuhalten ist, dass einzig und allein das implizite Euler-Verfahren als Basis für SDC A-stabil ist.
Zudem ist eine wichtige Erkenntnis, dass die Stabilitätsgebiete durch eine Aufwandssteigerung in Form einer Erhöhung der Anzahl der Quadratur-Knoten M und der Anzahl der Korrekturschritte K wachsen. Dies ist ein wesentliches Merkmal von SDC. Die oft eingesetzten klassischen Verfahren, wie das Adams-Bashforth-Verfahren, das Adams-Moulton-Verfahren, das BDF-Verfahren und das explizite Runge-Kutta-Verfahren, bieten dem Anwender ebenfalls Möglichkeiten eine inhärente Aufwandssteigerung durchzuführen. Bei dem Adams-Bashforth- bzw. Adams-Moulton-Verfahren und auch beim BDF-Verfahren[55] kann man z. B. weitere Stützstellen aus der Vergangenheit einbeziehen, also die Anzahl der vorgegebenen Startwerte erhöhen. Beim Runge-Kutta-Verfahren kann durch eine Erhöhung der Stufenzahl eine höhere Ordnung des Verfahrens erzielt werden. Aber die Aufwandssteigerungen sowohl beim Adams-Bashforth-Verfahren als auch beim Adams-Moulton-Verfahren und beim BDF-Verfahren und beim Runge-Kutta-Verfahren bewirken keine nennenswerte Vergrößerung der resultierenden Stabilitätsgebiete in der linken komplexen Halbebene. Sie können sogar zu einer Verkleinerung des Stabilitätsbereichs führen. Im Anhang A.1 sind die Stabilitätsgebiete der vier genannten Verfahren abgebildet und im Detail diskutiert.

[55] 'BDF' ist eine Abkürzung für Backward Differentiation Formulas.

Nachdem in diesem Kapitel also die einzelnen Stabilitäts- und Genauigkeitsgebiete für die einfache, allgemeine Testgleichung aufgezeigt und analysiert wurden, folgt im nächsten Kapitel der Arbeit die Analyse von Spectral Deferred Corrections angewandt auf die Slow-Wave-Fast-Wave-Gleichung, einer etwas abgewandelten Form der einfachen Testgleichung.

4. Vergleich der Verfahren für die Slow-Wave-Fast-Wave-Gleichung

In diesem Kapitel wird die Qualität des soeben erläuterten SDC-Verfahrens im Vergleich zur Qualität des zu Beginn der Arbeit angewendeten Trapez-Leapfrog-Verfahrens und des bereits erwähnten Adams-Bashforth-Verfahrens herausgestellt. In der Motivation, Kapitel 1, wurde beobachtet, dass das Trapez-Leapfrog-Verfahren und das Adams-Bashforth-Verfahren bei relativ geringem Aufwand einen großen relativen Fehler bei der Lösung der SWFW-Gleichung aufweisen. Bei dem nun folgenden detaillierten Vergleich der beiden klassischen Lösungsverfahren, wobei das Trapez-Leapfrog-Verfahren in der Praxis häufiger innerhalb eines semi-impliziten Lösers verwendet wird, mit dem neu eingeführten SDC-Verfahren werden die Stärken und Schwächen der numerischen Algorithmen per erneuter Betrachtung einer semi-impliziten Problemstellung herausgestellt.

Die Differentialgleichung, die eine schnelle Dynamik und eine langsame Dynamik umfasst – wie in Kapitel 1.2 erläutert –, ist gegeben durch

$$u'(t) = (\imath w_H + \imath w_L) \cdot u(t) = \imath w_H \cdot u(t) + \imath w_L \cdot u(t) \tag{123}$$

mit exakter Lösung $u(t) = e^{(\imath w_H + \imath w_L) \cdot t}$ zum Anfangswert $u(0) = 1$. w_H und w_L sind dabei die Frequenzen, die die Ausbreitung und Oszillation der schnellen und langsamen Welle beschreiben. Die Oszillationsfrequenz der langsamen Welle wird mit w_L und die der schnellen Welle mit w_H bezeichnet. Sinnvoll wäre nun eine Aufspaltung der Differentialgleichung und ein teilweise separiertes Lösungsverfahren, so dass w_L explizit und w_H implizit gelöst werden kann. Genau das ist die Vorgehensweise des semi-impliziten SDC-Verfahrens.

Nachfolgend werden zunächst das Trapez-Leapfrog-Verfahren und das Adams-Bashforth-Verfahren auf Stabilität hinsichtlich der SWFW-Gleichung untersucht und dann dem Stabilitätsgebiet von SDC mit unterschiedlich vielen Quadratur-Knoten und Korrekturschritten gegenübergestellt, da die Stabilität ein wesentliches Qualitätsmerkmal eines numerischen Verfahrens zur Lösung von Anfangswertproblemen darstellt.

4.1. Stabilitätsanalyse

Stabilität ist eine wichtige Eigenschaft eines Algorithmus zur numerischen Lösung von Differentialgleichungen. Durch sie werden die Auswirkungen von Rundungs- und Approximationsfehlern innerhalb des Lösungsverfahrens beschrieben. Nicht stabile Lösungen sind z. B. beim expliziten Euler-Verfahren angewandt auf die Testgleichung $u(t) = \lambda u(t)$ mit $u(0) = 1$ an der um die exakte Lösung nahezu oszillierenden, zackigen Form zu erkennen und daher nicht erwünscht. Generell bedeutet Instabilität einer Lösung, dass auftretende Fehler beliebig verstärkt werden. Die Fehlerverstärkung durch die Ausführung des numerischen Verfahrens soll möglichst minimal oder – noch besser – nicht vorhanden sein.

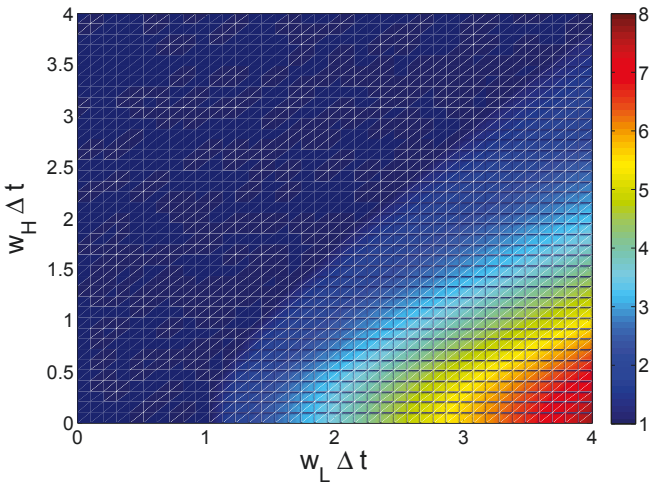


Abbildung 16: Amplifikationsfaktor beim Trapez-Leapfrog-Verfahren

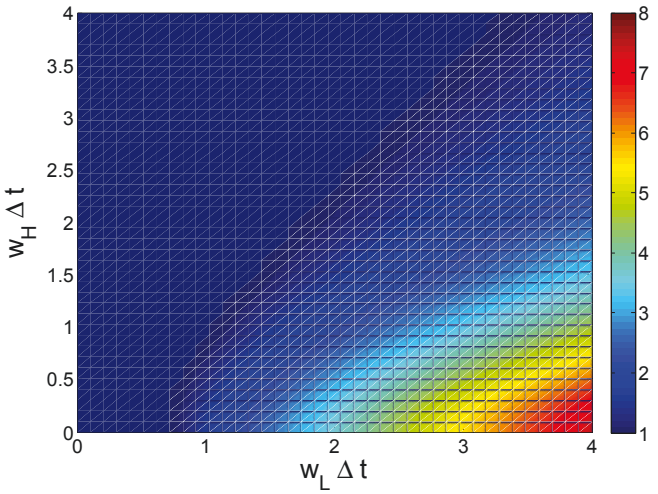


Abbildung 17: Amplifikationsfaktor beim Adams-Bashforth-Verfahren

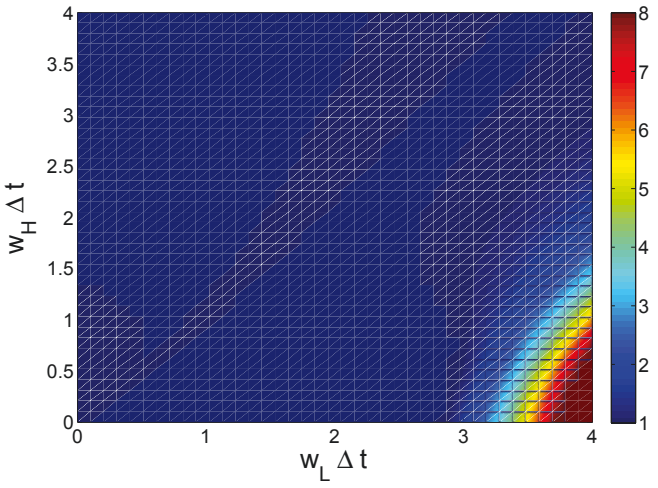


Abbildung 18: Amplifikationsfaktor beim SDC-Verfahren mit $M = 3$, $K = 4$, $N = 1$

Abbildung 16 zeigt den Amplifikationsfaktor des Trapez-Leapfrog-Algorithmus angewandt auf die SWFW-Gleichung. Der Amplifikationsfaktor berechnet sich im Allgemeinen wie bereits in Gleichung (119) erläutert. Ein numerisches Verfahren heißt stabil, wenn für den Amplifikationsfaktor $|\,Am(\lambda)\,| \leq 1$ gilt. In diesem Fall liegt keine nennenswerte Fehlerverstärkung in der Auswertung des Verfahrens innerhalb eines Zeitschrittes vor.

Die oberste Abbildung 16 zeigt den Amplifikationsfaktor für das Trapez-Leapfrog-Verfahren angewandt auf die SWFW-Gleichung mit variierendem $w_H\Delta t$ und $w_L\Delta t$. Der gewünschte Amplifikationsfaktor von $|\,Am(\lambda)\,| \leq 1$ ist nur in einem kleinen Bereich um die y-Achse festzustellen.[56] Das bedeutet, dass der Trapez-Leapfrog-Algorithmus für große $w_L\Delta t$, d. h. für große Schwingungsfrequenzen im expliziten Lösungsteil der Differentialgleichung, nicht stabil ist. Groß ist hier relativ zu betrachten, denn bereits bei einem $w_L\Delta t$ unwesentlich größer als 1 kommt der Trapez-Leapfrog-Algorithmus an seine Stabilitätsgrenze bei einem gleichzeitig nicht existierenden impliziten Anteil der Gleichung mit $w_H\Delta t = 0$. Bei wachsendem $w_H\Delta t$ wird auch die Toleranzgrenze für größere $w_L\Delta t$ höher, so dass die Stabilitätsgrenze ab $(1.5, 1.5)$ nahezu eine Winkelhalbierende des ersten Quadranten ist. Der sonst schnell zu Instabilitäten neigende Trapez-Leapfrog-Algorithmus ist bei einem nicht existenten expliziten Gleichungsanteil immer stabil, unabhängig davon, wie groß das $w_H\Delta t$ gewählt wird. Dies ist selbstverständlich, da es sich um ein semi-implizites Verfahren handelt.

In der mittleren Abbildung 17 ist der Amplifikationsfaktor für das Adams-Bashforth-Verfahren angewandt auf die SWFW-Gleichung eingezeichnet. Es ist nahezu identisch mit dem Stabilitätsgebiet für das Trapez-Leapfrog-Verfahren in Abbildung 16. Die einzige kleine Abweichung wird in dem Farbübergang von dunkelblau zu hellblau ersichtlich. Bei dem Adams-Bashforth-Verfahren ist dieser Bereich größer. Dies bedeutet, dass die Stabilitätsgrenze breiter ist, dass also mehrere Amplifikationsfaktoren sehr nahe am Wert 1 liegen.

Die Abbildung 18 zeigt das Stabilitätsgebiet des unter gleichen Voraussetzungen angewandten SDC-Verfahrens mit drei Quadratur-Knoten und vier Korrekturschritten. Auf den ersten Blick ist zu erkennen, dass der dunkelblaue Bereich, in dem der Amplifikationsfaktor sehr nahe bei 1 liegt bzw. gleich dem Wert 1 ist, sehr viel größer ist. Nur in der Ecke unten rechts bei großem $w_L\Delta t$ und kleinem $w_H\Delta t$ sind Instabilitäten festzustellen. Dieser instabile Bereich ist von der Lage und Form her mit dem des Trapez-Leapfrog-Verfahrens und dem des Adams-Bashforth-Verfahrens vergleichbar. Jedoch ist er deutlich kleiner. Dass genau bei der gerade angesprochenen Konstellation der Frequenzanteile Probleme bei der numerischen Lösung auftreten, liegt daran, dass w_L die Frequenz der langsamen Welle ist und explizit gelöst wird. Wird der explizite Lösungsanteil hochfrequenter, so wird das Verfahren instabil.

[56] Da sich die Stabilitätsgebiete symmetrisch um die y-Achse verhalten, ist auf die Darstellung des negativen x-Achsenbereichs verzichtet worden.

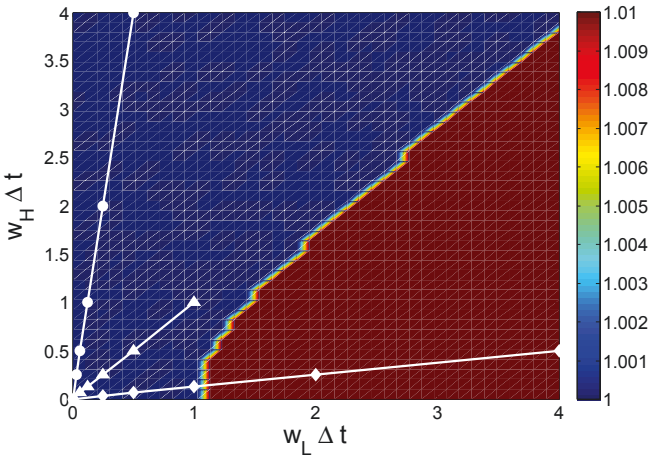


Abbildung 19: Amplifikationsfaktor beim Trapez-Leapfrog-Verfahren, abgebildet für den Bereich $1 <\mid Am(\lambda) \mid < 1.01$

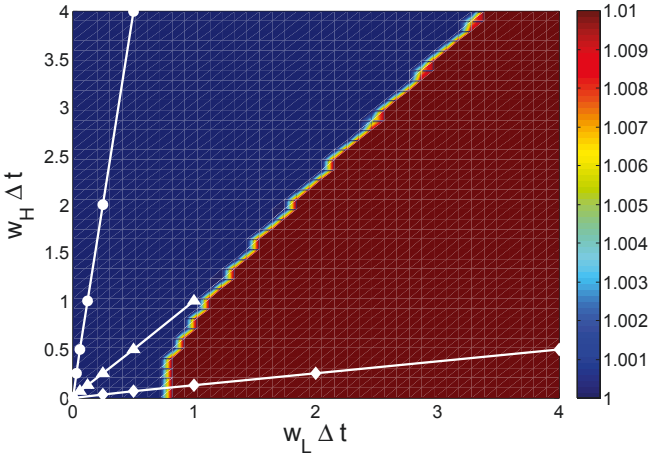


Abbildung 20: Amplifikationsfaktor beim Adams-Bashforth-Verfahren, abgebildet für den Bereich $1 <\mid Am(\lambda) \mid < 1.01$

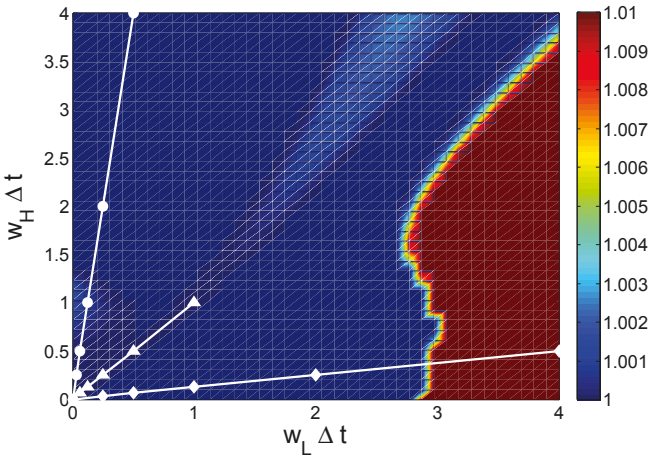
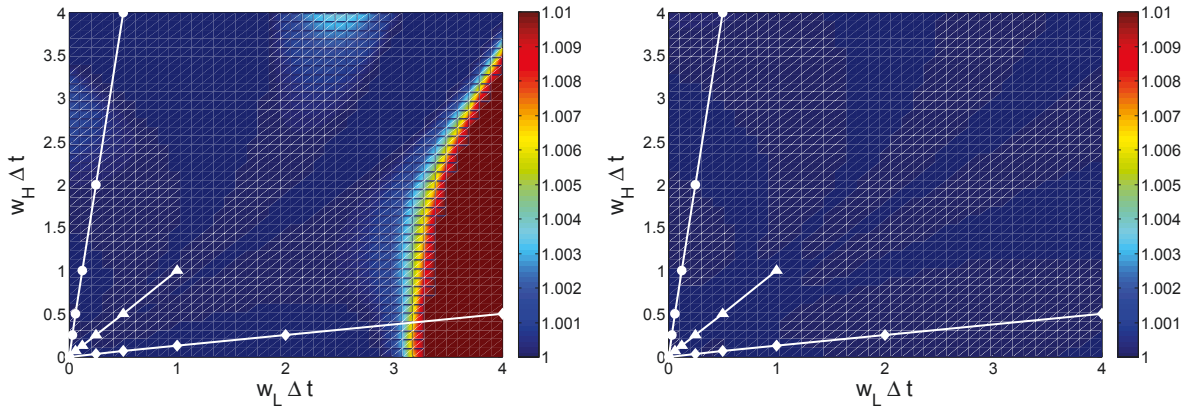


Abbildung 21: Amplifikationsfaktor beim SDC-Verfahren mit $M = 3$, $K = 4$, $N = 1$, abgebildet für den Bereich $1 <\mid Am(\lambda) \mid < 1.01$

Um einen besseren Vergleich zur Publikation von Dutt et al. [31] herstellen zu können, wird in den Abbildungen 19, 20 und 21 nur der Bereich $1 <| Am(\lambda) |< 1.01$ in den Visualisierungen farblich dargestellt. Die dort in weiß eingezeichneten Geraden werden im nächsten Abschnitt der Arbeit, im Unterkapitel 4.2, im Detail erläutert und zur weiteren Analyse der numerischen Verfahren verwendet. Allgemein ist noch anzufügen, dass die gerade benannten Abbildungen interpolierte Werte zur Darstellung nutzen, um die Sichtbarkeit einzelner Raster und Pixel zu vermeiden. Durch diese Interpolation kommt auch das Farbspektrum, ähnlich einem Regenbogen, am Farbübergang der beiden vorkommenden Farbbereiche – von blau zu rot – zustande.

Die Abbildung 19 ist äquivalent zur eben erläuterten Abbildung 16. Genauso wie die Abbildung 20 äquivalent zur eben erläuterten Abbildung 17 ist. Der hier blau gefärbte Bereich kennzeichnet das Stabilitätsgebiet des Trapez-Leapfrog-Algorithmus bzw. des Adams-Bashforth-Algorithmus.
Die Abbildung 21 ist äquivalent zur soeben erläuterten Abbildung 18. Auch in dieser Darstellung ist der blaue Bereich derjenige, der für Stabilität des Verfahrens steht. Die hellblauen Farbnuancen entstehen durch numerische Ungenauigkeiten und sind für die eigentliche Stabilität von SDC nicht relevant. Die bunten Farbanteile an der Stabilitätsgebietsgrenze sind ebenfalls auf numerische Ungenauigkeiten und die interpolierte Darstellungsweise zurückzuführen. Aussagen über die Stabilität auf diesem besagten Rand sind anhand von dieser Auswertung nicht definitiv zu treffen.

Weiterhin werden an dieser Stelle andere Varianten des SDC-Verfahrens hinsichtlich ihrer Stabilität untersucht.



(a) Amplifikationsfaktor beim SDC-Verfahren mit $M = 5, K = 8, N = 1$

(b) Amplifikationsfaktor beim SDC-Verfahren mit $M = 7, K = 12, N = 1$

Abbildung 22: Stabilitätsgebiete von SDC ($M = 5$, $K = 8$) und SDC ($M = 7$, $K = 12$) im Bereich $1 <| Am(\lambda) |< 1.01$

In Teil (a) von Abbildung 22 ist das semi-implizite SDC-Verfahren mit fünf Quadratur-Knoten zur Lösung der SWFW-Gleichung mit variierendem $w_H \Delta t$ und variierendem $w_L \Delta t$ untersucht worden. Dabei ist ein noch größeres Stabilitätsgebiet festzustellen, wobei es bei den glei-

chen Zusammenstellungen von impliziten und expliziten Gleichungsanteilen zu Instabilitäten kommt wie bei SDC mit nur drei Quadratur-Knoten. Eine weitere Vergrößerung des Stabilitätsgebiets bezüglich dieses Anfangswertproblems wird durch die Verwendung von noch mehr Quadratur-Knoten beim semi-impliziten SDC-Verfahren erreicht. Teil (b) von Abbildung 22 lässt leicht erkennen, dass SDC mit $M = 7$, also sieben Quadratur-Knoten, bei allen möglichen Kombinationen von $w_H\Delta t \in [0;4]$ und $w_L\Delta t \in [0;4]$ stabil ist. Es gibt in diesem Bereich keine nicht blau gefärbte Fläche.

SDC hat, wie gerade gezeigt, Vorteile hinsichtlich der Stabilität bei der Lösung der SWFW-Gleichung auf dem Intervall von 0 bis 1 für den Bereich $w_H\Delta t \leq w_L\Delta t$. Der Parameterbereich oberhalb der Winkelhalbierenden in den gerade betrachteten und analysierten Stabilitätsabbildungen – $w_H\Delta t \geq w_L\Delta t$ – ist jedoch der für die SWFW-Gleichung relevante Bereich, da w_H den schnellen Gleichungsanteil und w_L den langsamen Gleichungsanteil darstellt. In dem gerade angesprochenen, relevanten Bereich sind sowohl das Trapez-Leapfrog-Verfahren und das Adams-Bashforth-Verfahren, als auch die Varianten des SDC-Verfahrens stabil.
Das SDC-Verfahren höherer Ordnung, so wie es hier zur Auswertung des Anfangswertproblems verwendet wurde, erzeugt also größere Stabilitätsgebiete. Davon ist bei allgemeinen Mehrschrittverfahren nicht auszugehen. Nach der zweiten Dahlquist-Barriere gilt, dass es kein A-stabiles Mehrschrittverfahren mit m Schritten gibt, so dass seine Ordnung $m > 2$ ist [18]. Bei der Konstruktion von Mehrschrittverfahren – wie z. B. den Adams-Verfahren – mit höherer Ordnung sind also Einbußen hinsichtlich der Stabilität hinzunehmen, obwohl ein zusätzlicher Aufwand zur Erzeugung nötig ist. Weitere Quellen der Instabilität sind zusätzlich benötigte Anfangswerte, die unter Umständen durch numerische Näherungsverfahren berechnet werden müssen [24].
Im Gegenteil dazu erzeugt SDC mit höherer Ordnung, wie bereits erwähnt, größere Stabilitätsgebiete und ist trotzdem in der Implementierung nicht aufwendiger. Es ist einfach die Anzahl der Quadratur-Knoten und der Korrekturschritte entsprechend zu verändern, um eine höhere Ordnung zu erhalten. Dies ist ein großer Vorteil des SDC-Verfahrens gegenüber den Mehrschrittverfahren.

Es stellt sich jetzt die Frage, welche Kosten, auch bei SDC, zur Gewährleistung der Stabilität in Kauf genommen werden müssen. Kosten sind in diesem Fall eine Frage der Funktionsauswertungen der Rechte-Seite-Funktion der Differentialgleichung bzw. des Aufwandes des SDC-Verfahrens, welcher mit wachsender Anzahl an Quadratur-Knoten ansteigt.

4.2. Genauigkeiten und Kosten

Zur Auswertung und zum Vergleich der Genauigkeit der Lösung resultierend aus dem SDC-Verfahren, Trapez-Leapfrog-Verfahren und dem Adams-Bashforth-Verfahren wird der relative Fehler betrachtet. In der scheinbar intuitiven Herangehensweise dieser Analyse würde ein Aufwandsvergleich basierend auf der genutzten Schrittweite Δt durchgeführt werden. Dies ist jedoch ein trügerischer Rückschluss, da SDC innerhalb eines Funktionsaufrufs mehrere Funktionsauswertungen der Rechte-Seite-Funktion durchführt.
In einer Iteration des SDC-Verfahrens werden im Allgemeinen $(M + (M \cdot K))$ Auswertungen

der Rechte-Seite-Funktion benötigt. Das bedeutet, dass sich die Anzahl der Funktionsauswertungen durch

$$AnzFunktionsauswertungen = (M + (M \cdot K)) \cdot AnzZeitschritte \tag{124}$$

berechnen lässt. Diese Gleichung kommt dadurch zustande, dass die Lösung der Testfunktion[57] zunächst an dem pro Iteration aktuellen Zeitpunkt t ausgewertet und auf die vorhandenen M Quadratur-Knoten verteilt werden muss. Bei einem genauen Vorgehen müsste hier mit dem Basis-Propagator – dem Euler-Verfahren – die Lösung für die einzelnen Quadratur-Knoten nach vorne geschoben werden. Dies würde jedoch einen größeren Aufwand erzeugen und bringt für den Verlauf des restlichen Verfahrens keine nennenswerten Vorteile.
In dem mit der einfachen, zeitunabhängigen Rechte-Seite-Funktion betrachteten SDC-Verfahren könnte die Addition von M durch eine Addition von 1 ersetzt werden, da die Lösung der Testfunktion an allen Quadratur-Knoten aufgrund der Zeitunabhängigkeit identisch ist. Aus Gründen der Konsistenz wird im Vergleichsfall jedoch auf die allgemeine Formel mit der Addition von M Funktionsauswertungen zurückgegriffen.
Zudem wird in jeder Iteration bei jeder Auswertung der Korrekturgleichung – K Mal – die Testfunktion an den M Quadratur-Knoten ausgewertet. Dies muss pro Iteration von SDC auf die M Funktionsauswertungen aufaddiert werden. Entscheidend für die Aufwandsanalyse ist hier der Faktor K. Würde man z. B. die Startwerte an den Quadratur-Knoten beliebig setzen und somit die Addition von M Funktionsauswertungen auf das Produkt sparen, müsste ein weiterer Korrekturschritt durchgeführt werden, um die gleiche Konvergenzordnung zu erhalten. Somit würde sich die Ersparnis im Vorschritt durch das Inkrementieren von K aufheben und mit

$$AnzFunktionsauswertungen = (M \cdot (K+1)) \cdot AnzZeitschritte = (M + (M \cdot K)) \cdot AnzZeitschritte \tag{125}$$

würde die Anzahl benötigter Funktionsauswertungen identisch bleiben.

Um die Vergleichsanalyse durchzuführen, sind nachstehend drei Abbildungen zu finden, die mit entsprechend gewählten Parametern erzeugt wurden. Sowohl in Abbildung 23, als auch in den Abbildungen 24 und 25 kann man erkennen, dass bei dem Trapez-Leapfrog-Verfahren, bei dem Adams-Bashforth-Verfahren und bei dem SDC-Verfahren mit drei, fünf und sieben Quadratur-Knoten der relative Fehler mit wachsender Anzahl an Funktionsauswertungen bzw. kleiner werdender Schrittweite Δt abnimmt.
Auffallend ist in allen drei Abbildungen, dass jeweils alle fünf eingezeichneten Kurven auf der x-Achse unterschiedliche Startpunkte haben. Dies ergibt sich daraus, dass wir die Differentialgleichung über den Zeitraum von 0 bis 1 lösen und die Größe des ersten Zeitschrittes zu Beginn der Analyse mit $\Delta t = 0.5$ gewählt ist. Das Trapez-Leapfrog-Verfahren, als durchgezogene Linie in blau eingezeichnet, beginnt deshalb mit zwei Funktionsauswertungen.[58] Da das Adams-Bashforth-Verfahren, welches als gestrichelte Linie eingezeichnet ist, drei Startwerte benötigt und nicht nur einen, wie das Trapez-Leapfrog-Verfahren, kann hier erst eine erste

[57] Diese ist in diesem Fall die Rechte-Seite-Funktion.
[58] Dies berechnet sich aus der Differenz der Intervallgrenzen dividiert durch die Schrittweite: $\frac{1-0}{0.5} = 2$

Verfahrensauswertung bei einer kleineren Schrittgröße erfolgen. Bei dem SDC-Verfahren mit drei Quadratur-Knoten M beginnt die rote, mit Kreisen markierte Kurve auf der x-Achse erst bei dem Wert 30. Genau den Wert 30 bei einer Anzahl von einer Iteration erhält man mit der Gleichung (124).[59] Äquivalent zu dieser Rechnung ergibt sich für das SDC-Verfahren mit fünf bzw. sieben Quadratur-Knoten M mit Gleichung (124) ein Startwert auf der x-Achse von 90 bzw. 182.

Betrachtet wird also zur Aufwandsanalyse hinsichtlich der Genauigkeit die Anzahl der Funktionsauswertungen, die in den Abbildungen 23, 24 und 25 zu sehen sind.[60]
Bei der Auswertung der gerade aufgezählten Abbildungen kommt auch den weißen Linien aus den Abbildungen 19, 20, 21 und 22 eine Bedeutung zu. Entlang der dort pro Abbildung eingezeichneten drei Linien findet die Aufwandsanalyse nun statt.

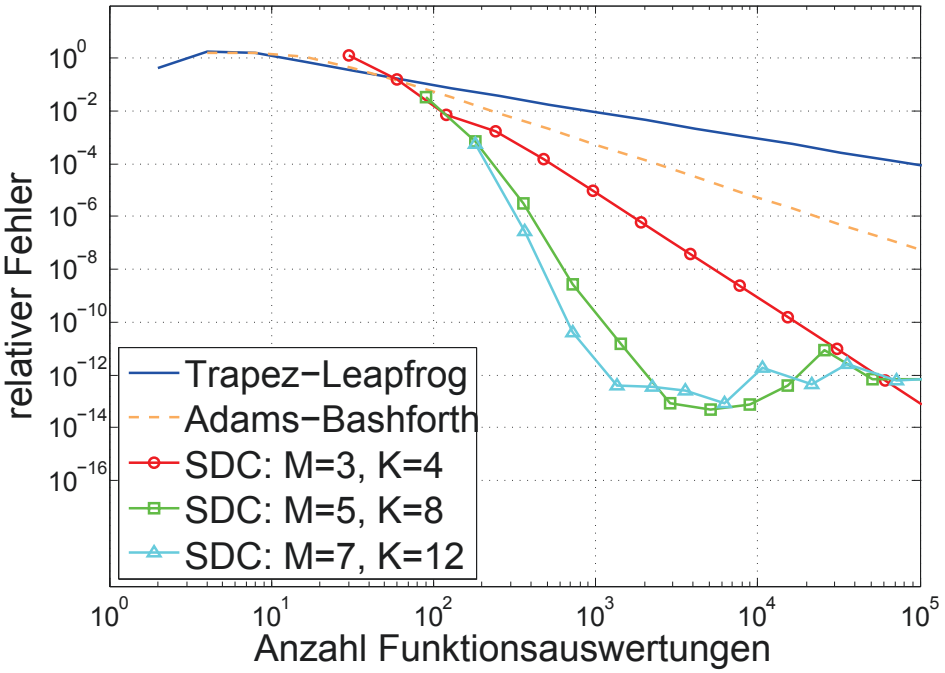


Abbildung 23: Aufwandsvergleich zwischen dem Trapez-Leapfrog-Verfahren, dem Adams-Bashforth-Verfahren, dem SDC-Verfahren mit $M = 3$, $K = 4$, dem SDC-Verfahren mit $M = 5$, $K = 8$, dem SDC-Verfahren mit $M = 7$, $K = 12$ zu $w_L = 0.5$ und $w_H = 4.0$

In Abbildung 23 ist jeweils entlang der obersten weißen Linie zwischen $w_L = 0.5$ und $w_H = 4$ analysiert worden. Diese Linie liegt, wie in den Abbildungen 19, 20, 21 und 22 ersichtlich ist,

[59] Es ergibt sich folglich die Zahl 30 aus $(3 + (3 \cdot 4)) \cdot 2$, wobei sich die Multiplikation mit zwei aus $\frac{1-0}{0.5} = 2$ ergibt, wie soeben beim Trapez-Leapfrog-Verfahren erläutert.

[60] In einem späteren Schritt wird noch einmal auf die mehr oder weniger 'unfaire' Analyse mittels der nötigen Schrittweite Δt eingegangen.

bei allen Lösungsmethoden in deren Stabilitätsgebiet und der Anteil der schnellen Schwingungswelle ist acht Mal so groß wie der Anteil der langsamen Schwingungswelle.
Entscheidend bei der Beurteilung der einzelnen Verfahren ist die unterschiedliche Steigung der Kurven und die daraus resultierenden Schnittpunkte. Die blaue Kurve ohne Marker zeigt den Verlauf des relativen Fehlers für das Trapez-Leapfrog-Verfahren. Diese fällt über die Anzahl an Funktionsauswertungen am geringsten ab. Beginnend bei einem Fehler um 1 erreicht man mit dieser Lösungsmethode selbst bei 10^5 Funktionsauswertungen nur einen relativen Fehler, der bei circa 10^{-4} liegt. Der Verlauf des relativen Fehlers beim Adams-Bashforth-Verfahren ist durch eine orangefarbene gestrichelte Linie eingezeichnet. Diese beginnt ebenfalls bei einem Fehler um den Wert 1 und fällt dann jedoch schneller ab als der relative Fehler beim Trapez-Leapfrog-Verfahren. Allerdings liegt der relative Fehler bei 10^5 Funktionsauswertungen noch über 10^{-8}. Um eine oft geforderte Grenze des relativen Fehlers von 10^{-5} zu unterschreiten, sind über 1000 Funktionsauswertungen nötig.[61]
Die anderen drei Kurven, die für den relativen Fehler verschiedener Varianten des semi-impliziten SDC-Verfahrens stehen, weisen einen steileren Verlauf auf. Hier wird die schon in Abbildung 6 gezeigte Konvergenzordnung entsprechend der Anzahl von Quadratur-Knoten noch einmal ersichtlich. Je mehr Quadratur-Knoten im SDC-Verfahren verwendet werden bei gleichzeitig wachsendem K, desto schneller fällt der relative Fehler ab. Die Ordnung des entsprechenden SDC-Verfahrens wächst also proportional zur Anzahl der Quadratur-Knoten mit entsprechender Anzahl an Korrekturschritten $K = 2M - 2$. Wie schon in Kapitel 3.1.2 zu sehen ist, hat SDC z. B. mit drei Quadratur-Knoten und vier Korrekturschritten die Ordnung 4. Eine wichtige Erkenntnis dieser Auswertung ist es, dass man mit beliebigen semi-impliziten SDC-Varianten eine wesentlich genauere Lösung der Differentialgleichung erhalten kann. Der relative Fehler ist bei einer genügend großen Anzahl von Funktionsauswertungen schnell kleiner als 10^{-12}. Ist der relative Fehler in dieser Größenordnung angekommen, bringen weitere Iterationen bzw. Funktionsauswertungen des SDC- Verfahrens hinsichtlich des relativen Fehlers keine nennenswerten Verbesserungen mehr. Das Niveau wird in etwa gehalten. SDC mit fünf Quadratur-Knoten und acht Korrekturschritten erreicht die eben schon angesprochene Grenze des relativen Fehlers von 10^{-5} schon bei circa der Hälfte der vom Trapez-Leapfrog-Verfahren benötigten Anzahl an Funktionsauswertungen.
Das Trapez-Leapfrog-Verfahren und das Adams-Bashforth-Verfahren sind jedoch nicht generell schlechter als die Varianten des semi-impliziten SDC-Verfahrens. Im Bereich der x-Werte von 1 bis kurz vor 100 liegen die Kurven unterhalb der Kurven, die von SDC generiert werden. Die Höhenlage der Kurven steht für die Größe des relativen Fehlers. Je niedriger eine Kurve liegt, desto exakter ist das Ergebnis, welches mittels dieses Verfahrens berechnet wurde. Das bedeutet, möchte man nur einen geringen Aufwand in die Lösung des Problems stecken, so ist auf jeden Fall entweder das Trapez-Leapfrog-Verfahren oder das Adams-Bashforth-Verfahren gegenüber einer semi-impliziten SDC-Variante vorzuziehen. Allerdings ist dann die Genauigkeit der generierten Lösung schlechter. Ebenfalls ist das Trapez-Leapfrog-Verfahren oder das Adams-Bashforth-Verfahren vorzuziehen, wenn nur eine ungenaue Näherungslösung gefragt

[61] Im Unterkapitel 1.2.1 wurde bereits mit Hilfe der Abbildung 3 auf die unterschiedlichen Verläufe des relativen Fehlers vom Trapez-Leapfrog-Verfahren und vom Adams-Bashforth-Verfahren hingewiesen und mit der unterschiedlichen Ordnung der Verfahren begründet.

ist. Dies ist z. B. bei Überschlagsrechnungen oder groben Lösungsbereichseingrenzungen der Fall. Mit nur 10 Funktionsauswertungen ist es selbst mit der semi-impliziten SDC-Variante mit $M = 3$ nicht möglich, überhaupt eine Lösung zu ermitteln.
Interessant sind weiterhin die Schnittpunkte der Kurven von den drei semi-impliziten SDC-Varianten mit den Fehler-Kurven des Trapez-Leapfrog-Verfahrens und des Adams-Bashforth-Verfahrens. Die Koordinaten dieser Schnittpunkte enthalten wichtige Informationen zur Vergleichsanalyse. Die x-Koordinaten geben an, mit wie viel Funktionsauswertungen der relative Fehler der beiden Verfahren, deren Kurven sich kreuzen, identisch ist. Die Information, die aus den y-Koordinaten der Schnittpunkte extrahiert werden kann, ist die Größe des relativen Fehlers, der mit exakt der gleichen Anzahl von Funktionsauswertungen der beiden Verfahren erzeugt wird. Anschaulich gesprochen bedeutet das: ist der relative Fehler der y-Koordinate eines Schnittpunkts gefordert, so macht es hinsichtlich des Aufwandes, gemessen an der Anzahl der Funktionsauswertungen, keinen Unterschied, welches der beiden Verfahren benutzt wird. Wird ein kleinerer Fehler als der im Schnittpunkt gefordert, so ist das Verfahren der nach dem Schnitt unterhalb verlaufenden Kurve zu bevorzugen. Das ist in Abbildung 23 in allen Fällen die semi-implizite SDC-Variante. Ist jedoch auch ein größerer relativer Fehler akzeptabel, so sollte das Verfahren der vor dem Schnittpunkt unterhalb verlaufenden Kurve gewählt werden. Dabei ist dann der relative Fehler kleiner, obwohl die Anzahl der Funktionsauswertungen geringer ist.
Konkret bedeutet das, dass die Varianten des semi-impliziten SDC-Verfahrens ab einem relativen Fehler von circa 10^{-1} besser sind als das Trapez-Leapfrog-Verfahren und das Adams-Bashforth-Verfahren. Der Aufwand, der durch die Ausführung von SDC anfällt, ist ab diesem Grenzwert kleiner als der Aufwand des Trapez-Leapfrog-Verfahrens und als der Aufwand des Adams-Bashforth-Verfahrens. Ab dem soeben erläuterten Schnittpunkt der Kurven verläuft die Linie des relativen Fehlers vom Adams-Bashforth-Verfahren unterhalb der Linie des relativen Fehlers vom Trapez-Leapfrog-Verfahren. Das heißt, dass ab diesem Zeitpunkt bei gleicher Anzahl an Funktionsauswertungen das Adams-Bashforth-Verfahren im Vergleich zum Trapez-Leapfrog-Verfahren genauere Ergebnisse bei der Auswertung der SWFW-Gleichung mit $w_L = 0.5$ und $w_H = 4.0$ liefert.
Im Vergleich der semi-impliziten SDC-Verfahren untereinander liegt die Kurve der Variante mit $M = 7$ und $K = 12$ im interessierenden Bereich stets unterhalb der anderen Kurven. Ein relativ kleiner Fehler ist somit mit einer vergleichsweise geringen Anzahl von Funktionsauswertungen zu erreichen. Für einen relativen Fehler von circa 10^{-12} werden etwas mehr als 1000 Funktionsauswertungen benötigt. Hier könnte jedoch die Verbesserung hinsichtlich des relativen Fehlers gestoppt werden, da dieser von diesem Zeitpunkt an nicht mehr an Größe verliert. Zusätzlicher Aufwand ist somit keine Garantie mehr für eine weitere Reduktion des Fehlers. Bei sieben Quadratur-Knoten wird bereits nach einer Verfahrensdurchführung der relative Fehler auf circa 10^{-3} reduziert. Jedoch ist die Anzahl der Funktionsauswertungen auch bereits nach einer Verfahrensdurchführung mit 182 Funktionsauswertungen ziemlich hoch. Es ist also mit dieser Variante nicht möglich eine sehr billige Approximation zu erzeugen.

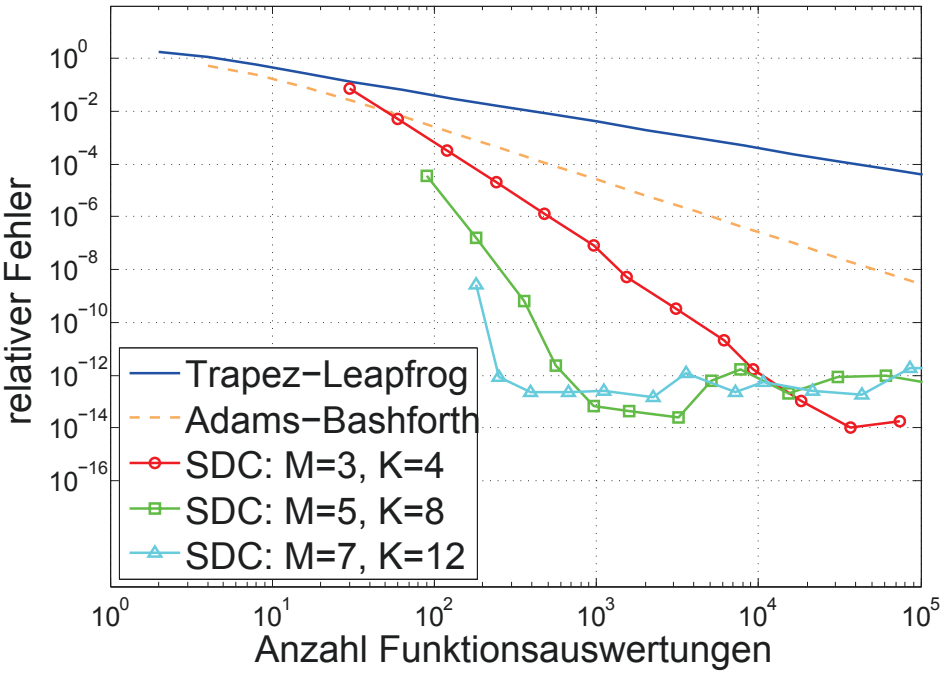


Abbildung 24: Aufwandsvergleich zwischen dem Trapez-Leapfrog-Verfahren, dem Adams-Bashforth-Verfahren, dem SDC-Verfahren mit $M = 3$, $K = 4$, dem SDC-Verfahren mit $M = 5$, $K = 8$, dem SDC-Verfahren mit $M = 7$, $K = 12$ zu $w_L = 1.0$ und $w_H = 1.0$

In einem zweiten Test wird nun die gleiche Aufwandsanalyse zwischen dem Trapez-Leapfrog-Verfahren, dem Adams-Bashforth-Verfahren und den semi-impliziten SDC-Varianten durchgeführt mit geänderten Parametern w_L und w_H. Abbildung 24 zeigt die entsprechenden Kurven der relativen Fehler für $w_L = w_H = 1$.
Betrachtet man zunächst die Lage dieser Parameter anhand der weißen Linien der Abbildungen 19, 20, 21 und 22 im Stabilitätsgebiet der einzelnen Verfahren, so wird ersichtlich, dass auch bei dieser Wahl der Parameter das Stabilitätsgebiet keiner Methode verlassen wird. Auf der Winkelhalbierenden bis zum Punkt $(1, 1)$ ist das Verfahren bezüglich aller möglichen Parametervariationen stabil.
Im Vergleich zur vorangehenden Analyse ist der Anteil der schnellen Schwingungswelle genauso groß wie der Anteil der langsamen Schwingungswelle. Zusätzlich wird durch den kleineren Wert des schnellen Anteils $w_H = 1$ die Geschwindigkeit der schnellen Welle reduziert. Konkret in dem analysierten Fall breiten sich die schnelle und langsame Welle mit gleicher Geschwindigkeit aus. Daraus erschließt sich, dass die Differentialgleichung nicht steif ist. Diese Tatsache verbessert den relativen Fehler in allen analysierten Verfahren.
Sowohl die Kurve des Trapez-Leapfrog-Verfahrens – blau, durchgezogen und ohne Marker –, als auch die Kurve des Adams-Bashforth-Verfahrens – orange und gestrichelt – zeigen einen steiler abfallenden Verlauf. Zudem ist der kleine Bogen beim Trapez-Leapfrog-Verfahren, der in den Abbildungen 23 und 25 bei geringer Anzahl von Funktionsauswertungen zu sehen ist, komplett verschwunden. Man kann hier tatsächlich von einem Geradenverlauf reden.

Der relative Fehler des Adams-Bashforth-Verfahrens liegt bereits von Beginn an unter dem des Trapez-Leapfrog-Verfahrens. Bei 10^5 Funktionsauswertungen ist der relative Fehler des Trapez-Leapfrog-Verfahrens von anfänglich circa 1 bis auf etwas mehr als 10^{-4} reduziert. Der relative Fehler des Adams-Bashforth-Verfahrens liegt bei 10^5 Funktionsauswertungen zwischen 10^{-8} und 10^{-9}. Damit ist er bei gleichem Aufwand eine Größenordnung kleiner als bei einer Parameterwahl von $w_L = 0.5$ und $w_H = 4$. Auch für die semi-impliziten SDC-Varianten scheint diese Parameterwahl mit gleichen, kleinen Wellenanteilen günstiger. Bereits nach einer Durchführung einer jeden semi-impliziten SDC-Variante ist der relative Fehler wesentlich kleiner als im zuvor analysierten Fall. Erkennbar ist diese Tatsache an den Startpunkten der einzelnen Kurven. Dieser liegt für SDC mit $M = 3$ und $K = 4$ deutlich unterhalb von 1. Für SDC mit fünf Quadratur-Knoten und sieben Korrekturschritten liegt der Fehler bereits unterhalb von 10^{-4}, was eine Größenreduktion – im Vergleich zum ersten Beispiel – um mehr als zwei Größenordnungen bedeutet und das schon nach nur einer Durchführung des SDC-Verfahrens. Ein noch niedriger gelegener Startpunkt ist bei SDC mit $M = 7$ und $K = 12$ zu erkennen. Hier ist der relative Fehler zu Beginn der Iterationen schon kleiner als 10^{-8}. Dies ist ein bewundernswertes Ergebnis, wenn man in Betracht zieht, dass im vorangehenden Beispiel der relative Fehler zu Beginn mehr als 10000 mal so groß ist. Die Steigung des Kurvenverlaufs bezüglich des relativen Fehlers der semi-impliziten SDC-Varianten ist im Vergleich zur Steigung des Kurvenverlaufs beim Trapez-Leapfrog-Verfahren und beim Adams-Bashforth-Verfahren nicht verändert. Der Abfall des Fehlers erfolgt nicht schneller, da die Konvergenzordnung durch eine andere Wahl der Parameter unverändert bleibt. Auch das Minimum des relativen Fehlers wird durch eine andere Parameterwahl bei den verwendeten semi-impliziten SDC-Varianten nicht kleiner. Es liegt wie auch in Abbildung 23 in einem Bereich zwischen 10^{-12} und 10^{-13}. Entscheidend ist jedoch, dass dieses Minimum bereits mit einer wesentlich kleineren Anzahl an Funktionsauswertungen erreicht werden kann. Bei SDC mit fünf Quadratur-Knoten und acht Korrekturschritten ist die besagte Sättigung des relativen Fehlers schon nach circa 1000 Funktionsauswertungen erreicht. Bei einer Parameterwahl von $w_L = 0.5$ und $w_H = 4$ werden dazu erheblich mehr Funktionsauswertungen benötigt. Das gleiche gilt für SDC mit $M = 7$ und $K = 12$.
Resultierend daraus, dass die Kurvenstartpunkte der Fehler der semi-impliziten SDC-Varianten niedriger positioniert sind, ist in Abbildung 24 nur ein Kurvenschnittpunkt zu sehen. Es schneiden sich die Kurven des relativen Fehlers des Adams-Bashforth-Verfahrens und des SDC-Verfahrens mit nur drei Quadratur-Knoten. Die x-Koordinate dieses Schnittpunkts liegt bei etwas weniger als 100 Funktionsauswertungen, was ungefähr der x-Koordinate des Schnittpunktes aus der voranstehenden Abbildung entspricht. Nennenswert verändert hat sich nur die y-Koordinate des Schnittpunkts. Diese liegt bei circa 10^{-2}, was genau einer kleineren Größenordnung des relativen Fehlers wie in Abbildung 23 entspricht. Das bedeutet, dass das semi-implizite SDC-Verfahren mit drei Quadratur-Knoten erst bei einem relativen Fehler von circa 10^{-2} mit geringerem Aufwand die Genauigkeit des Adams-Bashforth-Verfahrens unterschreitet. SDC ist also auch für nicht-steife Differentialgleichungen dem einfachen Trapez-Leapfrog-Verfahren und dem einfachen Adams-Bashforth-Verfahren zu bevorzugen, sobald eine Lösung mit einem relativen Fehler gefordert ist, der die Grenze von 10^{-2} unterschreitet.

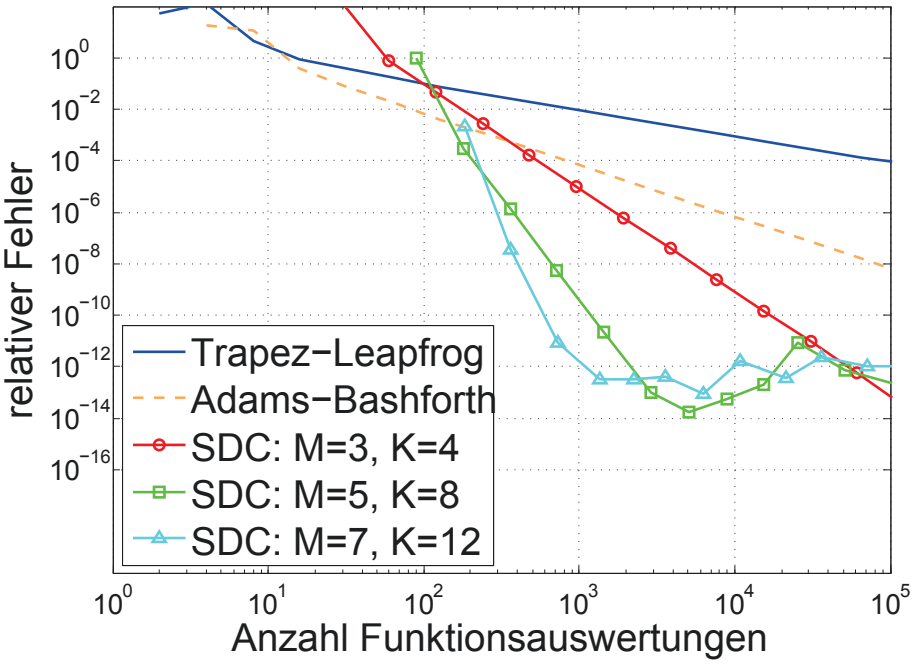


Abbildung 25: Aufwandsvergleich zwischen dem Trapez-Leapfrog-Verfahren, dem Adams-Bashforth-Verfahren, dem SDC-Verfahren mit $M = 3$, $K = 4$, dem SDC-Verfahren mit $M = 5$, $K = 8$, dem SDC-Verfahren mit $M = 7$, $K = 12$ zu $w_L = 4.0$ und $w_H = 0.5$

Abbildung 25 zeigt die dritte und letzte Analyse im Bezug auf Genauigkeit dieser Lösungsverfahren mit den Parametern $w_L = 4$ und $w_H = 0.5$. Schaut man sich auch hier die Lage dieser Parameter im Stabilitätsgebiet der einzelnen Verfahren mit Hilfe der eingezeichneten Linien in den Abbildungen 19, 20, 21 und 22 an, so erkennt man, dass die Geraden beim Trapez-Leapfrog-Verfahren, beim Adams-Bashforth-Verfahren und bei SDC mit drei und fünf Quadratur-Knoten den Stabilitätsbereich verlassen. Nur bei der semi-impliziten SDC-Variante mit $M = 7$ und $K = 12$ liegt der Punkt ($w_L = 4$,$w_H = 0.5$) noch innerhalb des Stabilitätsgebiets. Der Anteil des explizit gelösten Gleichungsterms ist hier mit dem Wert 4 acht Mal so groß wie der Wert des implizit gelösten Gleichungsanteils. Da die schnelle Welle einen impliziten Löser benötigt, um stabile Ergebnisse zu erhalten, kommt es bei dieser Parameterkombination in den meisten Fällen zu Stabilitätsproblemen. Bei $w_L = 4$ und $w_H = 0.5$ wird nämlich die sich schnell ausbreitende Welle explizit und die sich langsam ausbreitende Welle implizit behandelt.
Es ist nun zu untersuchen, in welchem Ausmaß diese Instabilität der drei genannten Lösungsverfahren die Größe des relativen Fehlers im Zusammenhang mit dem erzeugten Lösungsaufwand beeinflusst. Dazu wird Abbildung 25 im Vergleich mit Abbildung 23 betrachtet. In Abbildung 23 sind die gleichen Parameter gewählt worden. Jedoch wurde der schnelle Schwingungsanteil wirklich implizit und der langsame Schwingungsanteil explizit gelöst. Genau umgekehrt ist dies in Abbildung 25 realisiert worden.

Vergleicht man nun zuerst die blauen Kurven ohne Marker, die den Verlauf des relativen Fehlers des Trapez-Leapfrog-Verfahrens darstellen, miteinander so wird sofort klar, dass diese ab einem relativen Fehler von circa 0.1 identisch sind. Zuvor treten jedoch bei größerem w_L und kleinerem w_H Instabilitäten auf. Bei weniger als 10 Funktionsauswertungen hat die Kurve des Trapez-Leapfrog-Verfahrens eine starke Abweichung nach oben, also einen großen relativen Fehler. Ursächlich dafür ist die Instabiliät des Verfahrens für die SWFW-Gleichung mit $w_L = 4$ und $w_H = 0.5$. Erst bei kleineren Schrittweiten kann diese kompensiert werden und der relative Fehler wird gemäß der Ordnung des Verfahrens reduziert.
Das Adams-Bashforth-Verfahren zeigt ein ähnliches Verhalten, wie das Trapez-Leapfrog-Verfahren für große Schrittweiten. Auch hier sind vor allem in der Startphase des Verfahrens die Instabilitäten der Parameterwahl w_L und w_H sichtbar. Bei einer geringen Anzahl von Funktionsauswertungen ist der relative Fehler hier größer und das Verfahren konvergiert noch nicht. Dies macht sich auch im weiteren Verlauf der Kurve bemerkbar. Verwendet man bei der Lösung der SWFW-Gleichung mit dem Adams-Bashforth-Verfahren 1000 Funktionsauswertungen, so erhält man bei den Parametern $w_L = 0.5$ und $w_H = 4$ innerhalb der SWFW-Gleichung einen relativen Fehler von circa 10^{-3} und bei den Parametern $w_L = 4$ und $w_H = 0.5$ innerhalb der SWFW-Gleichung einen relativen Fehler von circa 10^{-4}. Die Vertauschung der Parameterwerte reduziert hier den relativen Fehler also um etwa eine Größenordnung.
Gleichermaßen zeigen die semi-impliziten Varianten des SDC-Verfahrens mit drei und fünf Quadratur-Knoten leichte Veränderungen beim Kurvenverlauf des relativen Fehlers aufgetragen zur Anzahl der Funktionsauswertungen auf. Auch dort wird das Stabilitätsgebiet bei großen w_L verlassen. Jedoch erst bei einem x-Wert im Bereich größer oder gleich 3. Die visuell auffälligste Veränderung ist der Startpunkt der roten Kurve mit kreisförmigen Markern und der grünen Kurve mit quadratischen Markern. Nach nur einem Durchlauf von SDC ist die resultierende Lösung also völlig unbrauchbar. Der Anfangspunkt von SDC mit $M = 3$ und $K = 4$ liegt sogar außerhalb der hier abgebildeten Grafik, was bedeutet, dass der relative Fehler hier anormal groß ist. Auch für SDC mit $M = 5$ und $K = 8$ liegt der Startpunkt bei einem y-Wert von 1. Ein relativer Fehler vom Wert 1 macht die Lösung absolut unbrauchbar. Im stabilen Fall von $w_L = 0.5$ und $w_H = 4$ ist der relative Fehler nur etwas größer als 10^{-2}. Hier macht sich die Instabilität dieser SDC-Variante bemerkbar. Da sich auch hier die Steigung der Fehlerkurven der semi-impliziten SDC-Varianten nicht ändert, bewirkt die Verschiebung des Startpunkts eine Verschiebung des ganzen Kurvenverlaufs. Bei gleichbleibendem x-Wert und unveränderter Fehlerkurve des Trapez-Leapfrog-Verfahrens verschiebt sich damit der Schnittpunkt dieser Kurven minimal nach rechts und nach unten. Das bedeutet, dass die Grenze, ab welcher die semi-impliziten SDC-Varianten gegenüber dem Trapez-Leapfrog-Verfahren zu bevorzugen sind, erst später überschritten wird. Erst bei einem kleineren relativen Fehler kann SDC mit weniger Aufwand als das Trapez-Leapfrog-Verfahren eine bessere Lösung erzeugen. Die Schnittpunkte der Kurven des relativen Fehlers der semi-impliziten SDC-Verfahren mit der Kurve des relativen Fehlers des Adams-Bashforth-Verfahrens verschiebt sich noch weiter nach rechts und nach unten als der gerade erläuterte Schnittpunkt mit dem Trapez-Leapfrog-Verfahren. Zuvor ist festgestellt worden, dass das Adams-Bashforth-Verfahren für $w_L = 4$ und $w_H = 0.5$, also für den instabilen Fall an Qualität einbüßt. Dennoch behält es im Fall von Konvergenz seine Konvergenzordnung von 2 bei. Da die semi-impliziten SDC-Verfahren noch stärker negativ von der Instabilität beeinflusst werden, reduziert sich die Grenze für den rela-

tiven Fehler, für dessen Erhalt das Adams-Bashforth-Verfahren gemessen an der Anzahl der Funktionsauswertungen zu bevorzugen ist.
Abschließend ist die Kurve des relativen Fehlers von SDC mit sieben Quadratur-Knoten zu erwähnen. Wie in Abbildung 22 zu sehen ist, liegen die Parameter hier innerhalb des Stabilitätsgebiets. Es macht also keinen Unterschied ob der Anteil mit $w_L = 4$ oder $w_L = 0.5$ implizit behandelt wird.[62] Der Kurvenverlauf der türkis farbigen Kurve mit dreieckigen Markern ist somit in Abbildung 23 und Abbildung 25 identisch.

Als Fazit kann man aus diesem Unterkapitel herausstellen, dass bei der Lösung der SWFW-Gleichung mit dem semi-impliziten SDC-Verfahren eine wesentlich genauere Lösung berechnet werden kann. Man sieht, dass dazu jedoch ein relativ großer Rechenaufwand von Nöten ist. Dieser ist jedoch ab einer gewissen Fehlergrenze, die sich je nach Parameterwahl verändert, kleiner als der Aufwand des Trapez-Leapfrog-Verfahrens und des Adams-Bashforth-Verfahrens um einen kleinen relativen Fehler zu erhalten. Es muss nun anhand der individuellen Problemstellung und Ansprüche an die Lösung abgewogen werden, welches Verfahren in welcher Situation zur Lösung der Differentialgleichung benutzt wird. Sobald ein kleiner relativer Fehler benötigt wird, ist SDC in jeder Hinsicht zu bevorzugen.

Um die Vorteile von SDC noch einmal zu verdeutlichen, wird an dieser Stelle die Anzahl der Funktionsauswertungen auf die genutzte Schrittweite Δt zurückgerechnet. Zu Beginn des Kapitels 4.2 wurde aus Gründen der Vergleichbarkeit direkt die Anzahl der Funktionsauswertungen als Maß für den Aufwand der Verfahren genutzt. An dieser Stelle wird nur zur Veranschaulichung der Dimensionen auf die Schrittgröße eingegangen.

In der Tabelle 3 ist beispielhaft der relative Fehler im Vergleich zu zwei Schrittweiten bezüglich vier der betrachteten Verfahrensvarianten zu den Parametern $w_H = 1$ und $w_L = 1$ aufgelistet.

Schrittweite	Verfahren	Anz. Funktions-auswertungen	relativer Fehler
0.01	Trapez-Leapfrog	100	$5.3321 \cdot 10^{-4}$
0.01	SDC: $M = 3, K = 4$	1500	$1.4221 \cdot 10^{-8}$
0.01	SDC: $M = 5, K = 8$	4500	$1.4998 \cdot 10^{-14}$
0.01	SDC: $M = 7, K = 12$	9100	$1.5031 \cdot 10^{-13}$
0.001	Trapez-Leapfrog	1000	$5.3333 \cdot 10^{-6}$
0.001	SDC: $M = 3, K = 4$	15000	$1.8745 \cdot 10^{-12}$
0.001	SDC: $M = 5, K = 8$	45000	$2.4905 \cdot 10^{-13}$
0.001	SDC: $M = 7, K = 12$	91000	$2.6171 \cdot 10^{-13}$

Tabelle 3: Aufwandsvergleich anhand der Schrittweiten für die Parameter $w_H = 1$; $w_L = 1$

[62] Und dementsprechend die Gleichungsanteile von w_H explizit behandelt werden.

Es ist ersichtlich, dass der Fehler bei einer Schrittweite von 0.01 einen Unterschied von bis zu zehn Größenordnungen aufweist. Betrachtet man hingegen die Anzahl der Funktionsauswertungen so relativiert sich dieser Abstand sehr schnell. Die Anzahl der Funktionsauswertungen variiert nämlich bei ein und derselben Schrittweite von 0.1 bis zu einem Faktor 91. Dies zeigt, dass auf der Basis der Schrittweite kein fairer Aufwandsvergleich durchgeführt werden kann. Für eine Schrittweite von 0.001 sind Ergebnisse mit ähnlicher Aussagekraft festzustellen.
Zusätzlich ist zu erwähnen, dass das Trapez-Leapfrog-Verfahren, um in einen Fehlerbereich entsprechend des semi-impliziten SDC-Verfahrens zu gelangen, die Schrittweite $\Delta t \approx 10^{-6}$ benötigt. Die Schrittweite von 10^{-6} steht hier einer Schrittweite bei SDC im Bereich von 10^{-3} gegenüber.

4.3. Zusammenfassung

Dieses Kapitel bildet einen Rahmen mit dem motivierenden Kapitel 1.2 über die SWFW-Gleichung. Im Vergleich zum vorangehenden Kapitel 3, in dem die numerischen Lösungsverfahren auf die allgemeine Testgleichung angewendet wurden, sind hier Stabilitäts- und Genauigkeitsuntersuchungen hinsichtlich der SWFW-Gleichung durchgeführt worden.
In Kapitel 1.2 dieser Arbeit wurde ersichtlich, dass das Trapez-Leapfrog-Verfahren und das Adams-Bashforth-Verfahren bei der Lösung der SWFW-Gleichung mit nicht ausreichend kleiner Schrittweite einen großen relativen Fehler aufweisen. Nur durch die Verwendung eines sehr kleinen Zeitschritts und damit verbunden vielen Auswertungen der Rechte-Seite-Funktion, kann der relative Fehler reduziert werden. Diese Aufwandssteigerung ist jedoch nicht immer wünschenswert und soll durch die Wahl eines alternativen Lösungsverfahrens unterbunden werden. Betrachtet man als alternativen Algorithmus das SDC-Verfahren, so erzielt man Verbesserungen hinsichtlich der Stabilität und der Genauigkeit. Diese Verbesserungen in Form einer Steigerung der Qualität der Lösung bringt gewisse Kosten mit sich.
Es resultiert die Frage, ob SDC bei der Lösung der SWFW-Gleichung generell und immer besser ist als das einfache Trapez-Leapfrog-Verfahren bzw. als das einfache Adams-Bashforth-Verfahren. Mit dem SDC-Verfahren kann eine wesentlich genauere Lösung der SWFW-Gleichung ermittelt werden. Dazu ist jedoch ein relativ großer Rechenaufwand von Nöten, der Kosten verursacht. Das Trapez-Leapfrog-Verfahren oder das Adams-Bashforth-Verfahren ist bei groben Näherungslösungen zu bevorzugen, da sie viel schneller und ohne großen Aufwand und damit verbundenen Kosten erste Approximationen liefern. Diese sind jedoch nicht so nah an der exakten Lösung. Der geringe Aufwand macht sich also sowohl in einer schlechteren Stabilität als auch in einer geringeren Rechengenauigkeit bemerkbar. Damit ist die finale Erkenntnis, dass SDC in jeder Hinsicht zu bevorzugen ist, sobald ein kleiner relativer Fehler benötigt wird.

Das nächste Kapitel ist ein Résumé der vorliegenden Arbeit und fasst die gewonnenen Erkenntnisse kurz zusammen. Ausgehend von diesem Erkenntnisstand können diverse weitere Analysen und Anwendungsuntersuchungen durchgeführt werden. Der abschließende Ausblick soll eine Idee vermitteln, auf welche Weise die Optimierung von SDC realisiert werden kann und in welchen anderen Anwendungsgebieten SDC sinnvoll eingesetzt werden kann.

5. Zusammenfassung und Ausblick

Als Ergebnis dieser Arbeit liegt eine detaillierte Analyse von Spectral Deferred Corrections angewandt auf das Slow-Wave-Fast-Wave-Problem vor.

Die Motivation stellte die gegebene Problemstellung des Lösens der SWFW-Gleichung dar. Schon Durran und Blossey analysierten einfache Verfahren, wie das Trapez-Leapfrog-Verfahren und das Adams-Bashforth-Verfahren, auf diese Problemstellung hin. Diese weisen jedoch Probleme hinsichtlich der Stabilität und Genauigkeit auf. In der vorliegenden Arbeit wurde ein neuartiges numerisches Lösungsverfahren eingeführt, womit stabile und genauere Ergebnisse der SWFW-Gleichung möglich sind.

5.1. Fazit

Dieses alternative Verfahren namens Spectral Deferred Corrections nutzt ein einfaches numerisches Basis-Verfahren – z. B. ein Euler-Verfahren – zur Bestimmung einer Approximation und verbessert diese Näherungslösung iterativ durch das Lösen einer Korrekturgleichung. Innerhalb dieser Korrekturgleichung gilt es ein Integral möglichst genau approximativ zu lösen. Dazu wird die Legendre-Gauß-Lobatto-Quadratur verwendet. Bei der Implementierung von SDC wird die iterative Vorgehensweise des Algorithmus ausgenutzt und die benötigten Gauß-Lobatto-Knoten werden über die Lösung eines Eigenwertproblems bestimmt.
Mit Hilfe dieser SDC-Implementierung in Matlab wurden im Rahmen dieser Arbeit zunächst die Ergebnisse von Dutt und Minion bezüglich der Konvergenz, Stabilität und Genauigkeit in mehreren Implementierungs-Varianten reproduziert. Im Anschluss daran wurde SDC auf die eigentliche Fragestellung hin angewendet und analysiert. Bei der Lösung der SWFW-Gleichung mittels Spectral Deferred Corrections konnten somit einige wichtige Erkenntnisse gewonnen werden.
Das Stabilitätsgebiet von SDC angewandt auf die SWFW-Gleichung ist wesentlich größer als das des einfachen numerischen Trapez-Leapfrog-Verfahrens bzw. des Adams-Bashforth-Verfahrens. Je aufwendiger das SDC-Verfahren konstruiert wird – bei Erhöhung der Anzahl der Quadratur-Knoten und/oder bei wachsender Anzahl an Iterationen –, desto größer wird das Stabilitätsgebiet. Doch hinter dieser Tatsache verbirgt sich auch der Nachteil von Spectral Deferred Corrections. Es kommt bei der Lösung des gegebenen Anfangswertproblems zu einer enormen Aufwandssteigerung und damit verbunden zu hohen Kosten. Eine große Anzahl an Iterationen wird bei SDC benötigt, um das eben angesprochene große Stabilitätsgebiet zu erzielen. Sinnvoll ist also eine Kosten-Nutzen-Analyse des SDC-Verfahrens angewandt auf die SWFW-Gleichung. Diese schließt sich in dieser Arbeit an die Stabilitäts- und Aufwandsvergleiche an und liefert die kommenden Endergebnisse.

Als Fazit ist schließlich herauszustellen, dass abhängig von den gewählten Parametern sowohl in der Ausführung des SDC-Algorithmus, als auch in der SWFW-Gleichung abzuwägen ist, mit welchem numerischen Verfahren die SWFW-Gleichung möglichst vorteilhaft zu lösen ist. Genaue Ergebnisse sind mit dem wenig aufwendigen und schnellen Trapez-Leapfrog-Verfahren bzw. Adams-Bashforth-Verfahren nicht sinnvoll zu realisieren. Nur grobe Nähe-

rungslösungen sind kostengünstig zu erhalten. Werden genaue Lösungen benötigt, ist SDC in jeglicher Hinsicht zu bevorzugen, da es schon ab einer relativ geringen Genauigkeitsanforderung kostengünstiger genauere Ergebnisse liefert als das Trapez-Leapfrog-Verfahren und das Adams-Bashforth-Verfahren.

Mit den aus dieser Arbeit resultierenden Ergebnissen kann nun in vielerlei Hinsicht weitergearbeitet werden. Eine Richtung der Weiterentwicklung besteht in der Effizienzsteigerung bzw. Parallelisierung von SDC in verschiedenen Varianten. Eine andere Richtung des Ausbaus der analysierten Problemstellung ist die Erweiterung der SWFW-Gleichung zu ähnlichen, aber wesentlich komplexeren Problemstellungen.

5.2. Ausblick

Die Spectral Deferred Corrections Methode kann in vielerlei Hinsicht eingesetzt und weiterentwickelt werden. Ein Nachteil von SDC, auf den auch in dieser Arbeit Bezug genommen wurde, ist der Aufwand und die damit verbundenen Kosten, die bei der Verwendung von SDC zur Lösung gegebener Anfangswertprobleme entstehen. Ausgehend von diesem Problemaspekt wurden einige Ansätze erforscht, die Lösung von Differentialgleichungen zu parallelisieren.
Bei gewöhnlichen Differentialgleichungen scheint dies jedoch auf den ersten Blick etwas verwunderlich, da diese mit der Zeit vorwärtsschreiten. Es scheint paradox, die weit entfernte Zukunft simulieren zu können, ohne die nahe Zukunft zu kennen. Um hier aber doch zeitparallel zu rechnen und das Zeitintervall, über welches das System betrachtet werden soll, auf mehrere Prozessoren aufzuteilen, entstand ein neuer 'parallel-in-time'-Ansatz. Dieser wird auch bei der Lösung von partiellen Differentialgleichungen eingesetzt, um diese sowohl im Raum als auch in der Zeit zu parallelisieren.

5.2.1. Parallel-in-time Integration

Die Basis eines ersten Ansatzes, der es ermöglichte SDC neben der Parallelisierung im Raum auch in der Zeit zu parallelisieren, war ein mehrstufiges SDC-Verfahren, das unter der Abkürzung 'MLSDC' bekannt wurde.[63] Bis jetzt wurde SDC immer auf einer Genauigkeitsstufe, auf einem gleichbleibenden Zeitgitter, betrachtet. Ein Ansatz basierend auf variierenden Zeitgittern wurde 2013 von Speck et al. [56] untersucht. Auf dieser Erweiterung des einstufigen SDC-Verfahrens auf ein mehrstufiges SDC-Vefahren, das bedeutet dem Lösen des Anfangswertproblems auf einer Hierarchie von Zeitgittern, also einer variierenden Anzahl an Kollokationspunkten, kann der sogenannte PFASST-Algorithmus aufbauen und mathematisch analysiert werden [56]. PFASST steht für ein 'parallel full approximation scheme in space and time'. Diese Methode setzt sich zusammen aus SDC, Parareal, hybridem Parareal/SDC und FAS [57].[64] FAS ist ein Mehrgitterverfahren für nicht-lineare Probleme. Es soll dabei eine

[63] MLSDC ist eine Abkürzung für 'A multi-level spectral deferred correction method' [56], ein mehrstufiges Spectral Deferred Corrections Verfahren.

[64] Die Methoden Parareal und hybrides Parareal/SDC werden im Laufe dieses Kapitels und im Anhang A.2 beschrieben.

Verringerung des Rechenaufwands stattfinden, indem der Fehler einer numerischen Approximation, welche auf einem feinen Gitter bestimmt wurde, auf einem gröberen Gitter berechnet wird [58].[65] Ein Genauigkeitsvergleich zwischen Runge-Kutta(4), SDC und PFASST angewandt auf ein Vortex-Modell wurde 2012 von Speck et al. durchgeführt [59]. Um die Vorgehensweise von PFASST etwas besser nachvollziehen zu können, befinden sich im Anhang A.3 dieser Arbeit zwei Grafiken, an denen die Funktionsweise von PFASST grob erläutert wird.

Durch die Verwendung verschiedener Zeitgitter bei der Lösung durch das mehrstufige SDC-Verfahren erfolgt eine Auswertung in verschiedenen Genauigkeitsstufen. Verwendet man ein grobes Zeitgitter, also eine der untersten Stufen der Genauigkeitshierarchie, so erhält man eine grobe Näherungslösung. Wird hingegen ein feines Zeitgitter bei der Auswertung von SDC benutzt, so werden genauere Näherungslösungen produziert. Auf diese Weise können z. B. bei einem zweistufigen SDC-Verfahren ein grober und ein feiner Propagator definiert werden, wobei der feine Propagator parallel ausgewertet werden kann.
Bei PFASST können also mehrere Zeitschritte parallel berechnet werden. Ein allgemeines Verfahren, in dem ebenfalls mehrere Zeitschritte parallel ausgewertet werden können, nennt sich Parareal. Dieser weitere parallel-in-time Lösungsansatz nutzt ebenfalls einen feinen und einen groben Propagator. Er wurde jedoch zunächst völlig unabhängig von SDC entwickelt und erste Überlegungen zu Parareal – einer zu dieser Zeit neuartigen Methode der parallelen Zeitdiskretisierung – wurde im Jahr 2001 von Lions, Maday und Turinici veröffentlicht [60].

Parareal ist ein Verfahren, welches – wie bereits angedeutet – unabhängig von SDC zur parallelen Zeit-Diskretisierung genutzt werden kann. Man verwendet, ähnlich wie bei einem zweistufigen MLSDC-Verfahren, einen feinen und einen groben Propagator, wobei der feine Propagator relativ genaue Ergebnisse liefert, dafür aber teuer in der Ausführung ist und der grobe Propagator nur grobe Näherungslösungen bestimmt, aber wenige Kosten mit sich bringt. Die Grundidee von Parareal liegt darin, dass mit Hilfe des groben Propagators, der meist nur auf einer groben Zeitauflösung seriell arbeitet, mehrere Anfangswerte für das zu betrachtende Zeitintervall generiert werden sollen. Basierend auf diesen, auf billige Weise seriell berechneten zusätzlichen Anfangswerten, kann dann der feine Propagator parallel für jedes Teilintervall eine genaue Lösung bestimmen. Der wichtige Punkt ist die parallele Ausführung des feinen Propagators. Er ist zwar teuer, doch durch die Möglichkeit der Parallelisierung über die Zeit, durch die vorher auf billige Weise bestimmten Anfangswerte für die Teilintervalle, fallen diese Kosten nicht seriell ins Gewicht. Nachdem also die Anfangswerte seriell mit einem groben Propagator berechnet wurden, kann die erste Parareal-Iteration erfolgen. Jede Parareal-Iteration besteht aus parallel laufenden Auswertungen, in denen die groben Anfangswerte mittels eines feinen Propagators propagiert werden, der Bestimmung einer neuen Näherung der Anfangswerte durch einen groben Propagator und der Verbesserung der Lösung durch die Verwendung der Anfangswerte aus der vorangehenden Parareal-Iteration, der gerade verbesserten Anfangswerte und der durch den feinen Propagator parallel ermittelten Lösung. An dieser Stelle wird zum besseren Verständnis der algorithmischen Grundidee auf den Anhang A.2 verwiesen. Dort wird mit Hilfe grafischer Ausgaben aus der Masterarbeit von Niel-

[65] Wie genau FAS im Zusammenhang mit PFASST funktioniert, dazu siehe [57].

sen [61] die Vorgehensweise visualisiert.
Parareal für gewöhnliche Differentialgleichungen wurde erstmals 2001 veröffentlicht [62]. Sowohl Lions, Maday, Turinici [60], als auch Nievergelt [63], Gander, Hairer [64], Gander, Vandewalle [65] und Staff, Ronquist [66] beschäftigten sich daraufhin tiefer mit der Theorie von Parareal, führten zusätzliche Konvergenzuntersuchungen durch und testeten das Verfahren an verbreiteten Testproblemen, wie der Lorenz-Gleichung oder der viscous Burger-Gleichung.
Wie schon erwähnt, kann Parareal auch zur Lösung von partiellen Differentialgleichungen verwendet werden [60]. Ferner wird die parallele Zeitdiskretisierung auch für nicht-lineare partielle Differentialgleichungen angewendet [67]. Ein Anwendungsbeispiel von Parareal für partielle Differentialgleichungen ist die parallele Integrationsmethode im Zusammenhang mit der langfristigen Entwicklung des Sonnensystems [68]. Auch in der Molekulardynamik sind zeitparallele Integrationsverfahren oft wünschenswert, wie z. B. bei mikroskopischen Simulationen über lange Zeiträume [69].

Im Jahr 2008 folgten dann neue Varianten des Parareal-Algorithmus, die Parareal in Verbindung mit Spectral Deferred Corrections brachten. Minion und Williams kombinierten das klassische Parareal mit SDC-Methoden und sprachen in diesem Zusammenhang von 'A Hybrid Spectral Deferred Corrections Method' [70]. Das bedeutet, dass der feine und der grobe Propagator aus Parareal mit SDC-Sweeps von unterschiedlicher Zeitauflösung zur Verbesserung der Lösung in jedem Schritt arbeiten. Es wird also die Deferred Corrections Strategie innerhalb der Parareal-Iterationen genutzt. Auch Minion erforschte dieses hybride Verfahren 2010, was auf Parareal mit klassischen Einschrittverfahren für den feinen und groben Propagator basiert [60]. Er unterteilte seine Analyse nach Parareal, Parareal-Deferred Corrections und hybrides Parareal-Spectral Deferred Corrections [71].

5.2.2. Acoustic-Advection System

Wie bereits in Kapitel 1.2 erwähnt, ist die SWFW-Gleichung eine besondere Variante der allgemeinen Testgleichung zur Analyse von numerischen Lösungsverfahren. Aus dieser einfachen SWFW-Testgleichung lassen sich komplexere Problemstellungen mit ähnlichen Eigenschaften herleiten.
Eine wichtige Anwendung basierend auf der SWFW-Gleichung ist das Acoustic-Advection System. Dies ist ein System partieller Differentialgleichungen, welches sich ebenfalls aus einer sich schnell und einer sich langsam ausbreitenden Welle zusammensetzt. Das System ist im Kontinuierlichen durch

$$u_t + c_s p_x + U u_x = 0 \tag{126}$$
$$p_t + c_s u_x + U p_x = 0 \tag{127}$$

gegeben [28]. Dabei ist c_s die Schallgeschwindigkeit und U eine mit der Zeit konstante Advektions-Geschwindigkeit [72]. Unbekannt ist das Geschwindigkeits-Störungs-Feld u und der Störungs-Druck p [72].
Es liegt mit dem Acoustic-Advection System ein steifes System an Differentialgleichungen vor, wenn $c_s \gg U$ ist. Dies ist typischer Weise in der Atmosphäre durch die Werte $c_s \approx 330 \frac{m}{s}$

und $U \approx 10\,\frac{m}{s}$ gegeben [28]. Die langsame Welle – äquivalent zur langsamen Welle der SWFW-Gleichung – ist hier durch die Summanden Uu_x und Up_x gegeben, die die Advektion beschreiben. Die Schallgeschwindigkeit, der sich schnell ausbreitenden Wellen, wird durch die Terme $c_s p_x$ und $c_s u_x$ berücksichtigt. Es ist also ein Lösungsverfahren wünschenswert, welches die Wellen von unterschiedlicher Geschwindigkeit separat und je nach den entsprechenden Anforderungen behandelt [28, 72]. Dies wird durch Ruprecht und Krause mittels eines zeit-parallelen Verfahrens realisiert [72]. Zur Lösung eines zweidimensionalen hyperbolischen Acoustic-Advection Systems verwenden sie eine modifizierte Parareal-Methode. Diese setzt sich zusammen aus einem dreistufigen Runge-Kutta-Verfahren als feiner Propagator und einem groben Propagator. Der grobe Propagator ist teilweise aufgespalten in das explizite Euler-Verfahren und die Forward-Backward-Methode, je nach Dynamik der Gleichungssystemanteile [72].
Bei der einfachen Übertragung von Parareal auf hyperbolische Probleme kommt es zunächst jedoch zu Problemen z. B. hinsichtlich der Stabilität. Doch schon Gander zeigte 2008, dass es bei einer Modifikation von Parareal effiziente Ergebnisse für Advektion auf begrenzten Gebieten gibt [73]. Diese modifizierte Parareal-Version wird auch von Ruprecht und Krause zur Analyse des Acoustic-Advection Systems genutzt.

Schließlich sind in der vorliegenden Arbeit Antworten auf die drei in der Motivation gestellten Fragen gefunden worden:
Es gibt ein numerisches Lösungsverfahren für Differentialgleichungen, wie die SWFW-Gleichung, die verschiedene Dynamiken enthalten, welches die separate Behandlung der Gleichungsteile realisiert. Diese Ansätze werden als semi-implizite Lösungsverfahren bezeichnet. Da dabei die Terme – je nach Dynamik – entweder explizit oder implizit gelöst werden können, ermöglichen semi-implizite numerische Verfahren eine Bestimmung von stabilen und genauen Ergebnissen für Probleme der atmosphärischen Wissenschaft, die ähnlich wie das SWFW-Problem sind.
Das zentrale Verfahren dieser Arbeit, welches oft in der semi-impliziten Implementierungsvariante genutzt wird, ist die Spectral Deferred Corrections Methode. Stabilität und Genauigkeit werden hier vor allem durch die spektrale Anordnung der Quadratur-Knoten innerhalb des Verfahrens gewährleistet. In der semi-impliziten Variante liefert SDC bei wachsender Anzahl an Quadratur-Knoten und Korrekturschritten weitestgehend stabile und genaue Ergebnisse für das SWFW-Problem mit verschiedenen Parametern.
Für diese genaueren Lösungen durch die Verwendung vieler Quadratur-Knoten und vieler Korrekturschritte, müssen allerdings zusätzliche Kosten in Kauf genommen werden. Doch um genaue Lösungen mit einem kleinen relativen Fehler zu erhalten, reicht es nicht aus, einfache Verfahren – wie das Trapez-Leapfrog- oder das Adams-Bashforth-Verfahren – zu verwenden. Der Aufwand, der bei der Verwendung dieser beiden Verfahren zur Lösung der SWFW-Gleichung bei einem geringen relativen Fehler anfällt, ist wesentlich größer, als die anfallenden Kosten durch die Lösung mittels des semi-impliziten SDC-Verfahrens.

SDC kann also eine stabile und genaue Lösung des SWFW-Problems mit adäquatem und akzeptablem Aufwand liefern.

A. Anhang

A.1. Stabilitätsgebiete von klassischen Lösungsverfahren

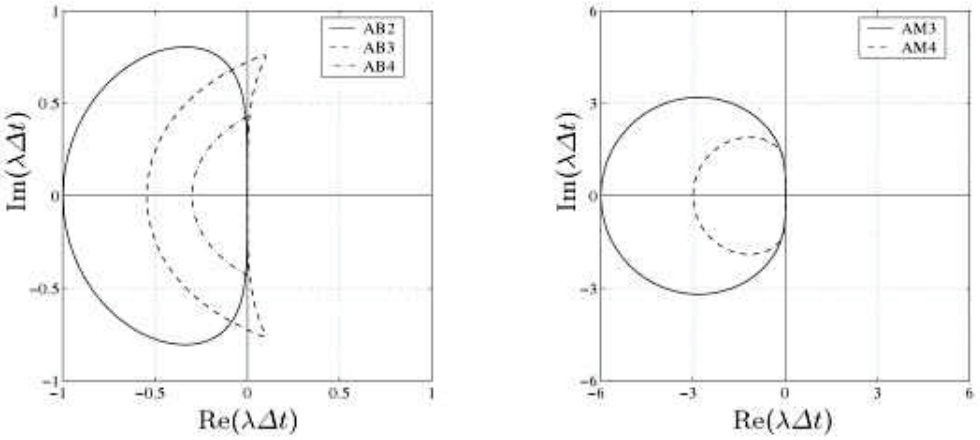


Abbildung 26: Stabilitätsgebiete des Adams-Bashforth- und des Adams-Moulton-Verfahrens mit variierenden Schrittzahlen [54]

Der linke Teil der Abbildung 26, welche aus dem Buch von Canuto, Hussaini, Quarteroni, Zang entnommen ist [54], zeigt die Stabilitätsgebiete des Adams-Bashforth-Verfahrens für die Schrittzahlen 2, 3 und 4. Es ist zu erkennen, dass sich die Stabilitätsgebiete in der negativen komplexen Halbebene bei wachsender Schrittzahl deutlich verkleinern und bei einer Schrittzahl von 4 sogar ganz verschwinden. Trotz des höheren Aufwandes reduziert sich der A-Stabilitätsbereich. In dem rechten Teil von Abbildung 26 sind die Stabilitätsgebiete für das dreistufige Adams-Moulton- und das vierstufige Adams-Moulton-Verfahren zu sehen. Auch hier reduziert sich das Stabilitätsgebiet bei wachsender Stufenanzahl. Das schrumpfende A-Stabilitätsgebiet bei zunehmender Stufenzahl für beliebige Mehrschrittverfahren wurde bereits von Dahlquist erforscht [18]. Hier weisen Mehrschrittverfahren einen großen Nachteil auf.

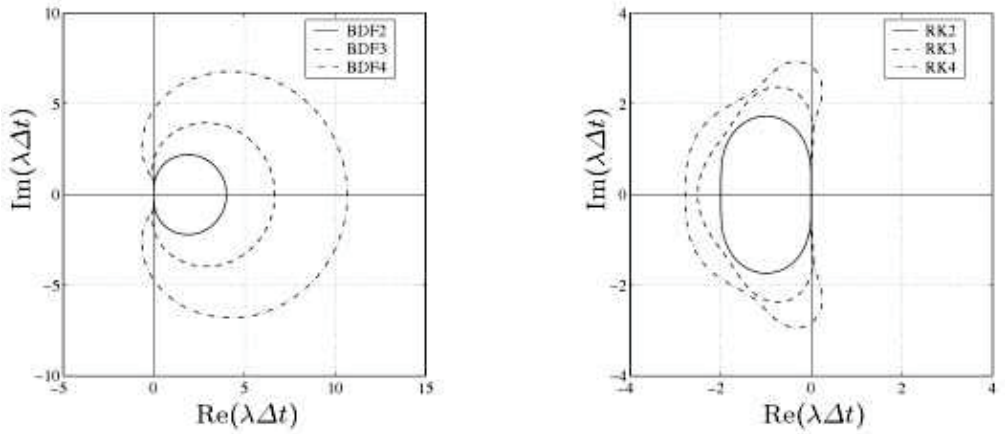


Abbildung 27: Stabilitätsgebiete des BDF- und des expliziten Runge-Kutta-Verfahrens mit variierenden Stufenzahlen [54]

In Abbildung 27, die ebenfalls aus dem Buch von Canuto et al. stammt [54], sind im linken Teil die Stabilitätsgebiete des zwei-, drei- und vierstufigen BDF-Verfahrens zu sehen. Diese wachsen zwar bei Erhöhung des Aufwands, aber nur ein minimaler Teil der zusätzlichen Fläche liegt in der negativen komplexen Halbebene. Daher ist die minimale Vergrößerung des A-Stabilitätsbereichs nahezu irrelevant. Die rechte Hälfte von Abbildung 27 zeigt die Stabilitätsgebiete des expliziten Runge-Kutta-Verfahrens mit zwei, drei und vier Stufen. Es ist eine Vergrößerung der Stabilitätsbereiche bei einer Erhöhung der Stufenzahl zu erkennen. Jedoch ist dies ein nicht nennenswertes Wachstum, welches im Vergleich zur Vergrößerung der Stabilitätsgebiete beim SDC-Verfahren durch eine Erhöhung der Anzahl an Quadratur-Knoten minimal ist. Zusätzlich fällt bei beliebigen Mehrschrittverfahren ein wesentlich größerer Rechenaufwand bzw. Implementierungsaufwand an, da die Bestimmung der Koeffizienten des Mehrschrittverfahrens einen zusätzlichen Aufwand erfordert [24]. Bei SDC kann die Vergrößerung des Stabilitätsgebiets durch eine einfache Erhöhung der Anzahl der Quadratur-Knoten und der Anzahl der Korrekturschritte innerhalb des Verfahrens realisiert werden. Damit entsteht kein größerer Implementierungsaufwand.

A.2. Parareal

In der Masterarbeit von Allan S. Nielsen [61] sind die in Abbildung 28 zu sehenden Plots abgebildet. Sie visualisieren auf anschauliche Weise das Konzept des Parareal-Algorithmus.

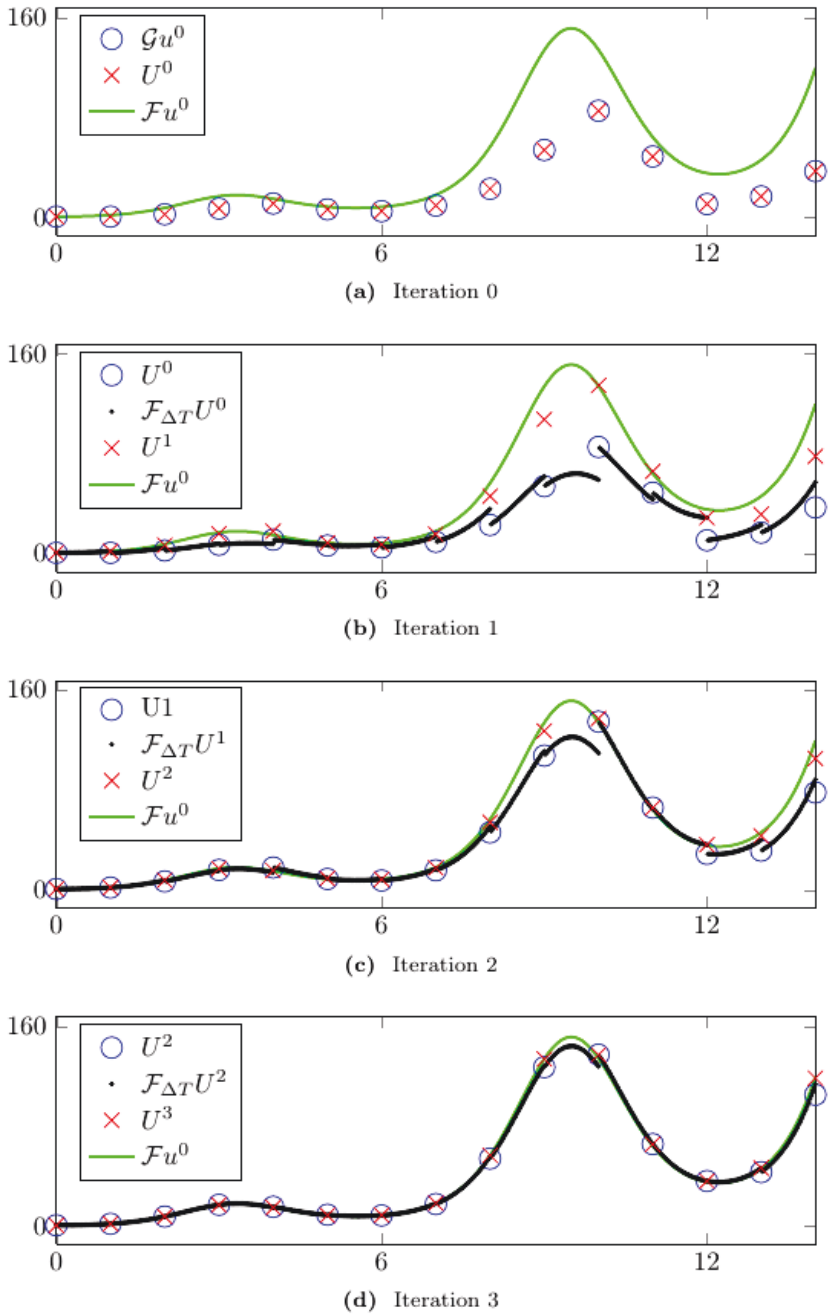


Abbildung 28: Visualisierung der Vorgehensweise von Parareal [61]

Zu sehen sind in Abbildung 28 die ersten vier Iterationen des Parareal-Algorithmus angewendet auf das spezielle Anfangswertproblem

$$y'(t) = sin(t) \cdot y(t) + t \quad (128)$$
$$y(0) = 1 \quad (129)$$

mit $t \in [0;14]$. Sowohl der grobe Propagator $\mathscr{G}_{\Delta T}$ als auch der feine Propagator $\mathscr{F}_{\Delta T}$ werden durch das implizite Euler-Verfahren realisiert, wobei der feine Propagator eine wesentlich kleinere Schrittweite verwendet und damit durch größeren Aufwand genauere Ergebnisse liefert [61].
In Teil (a) der Abbildung 28 ist der Initialisierungsvorgang sehr schön nachzuvollziehen. Die Vorhersage U^0 ist hier einfach das Ergebnis des groben Propagators $\mathscr{G}$ angewandt auf die Anfangsbedingung über den gesamten Zeitraum. Damit werden rein sequentiell grobe Anfangswerte für Teilintervalle von $[0;14]$ bestimmt, auf denen der feine Propagator aufsetzen kann. Die grüne, durchgehende Kurve ist die Lösung des Anfangswertproblems durch Verwendung des feinen Propagators rein sequentiell in der Zeit. Sie stellt die Referenzlösung dar.
In Teil (b) der Abbildung 28 wird aufbauend auf U^0 aus der vorangehenden Iteration eine neue korrigierte Vorhersage U^1 erzeugt. Dies geschieht durch Anwendung des feinen Propagators auf U^0. Diese Auswertungen durch den feinen und aufwendigen Propagator $\mathscr{F}$ können nun auf mehrere Prozessoren verteilt werden und somit parallel erfolgen. Auf jedem Prozessor kann unabhängig von den anderen ein Anfangswertproblem mit dem Anfangswert U^0 gelöst werden. Durch diese Lösungen der künstlich erzeugten Menge von Anfangswertproblemen durch den feinen Propagator können die sequentiell generierten Vorhersagen des groben Propagators im nächsten Schritt korrigiert werden. Auf diese Weise werden neue Startwerte U^1 für $\mathscr{F}$ auf jedem Teilintervall für die nächste Iteration bestimmt [61]. Die grüne Linie ist unverändert zum Abschnitt (a) dieser Abbildung die Referenzlösung.
Teil (c) und (d) von Abbildung 28 zeigen die zweite und dritte Parareal-Iteration, die auf gleiche Weise zu interpretieren sind, wie die erste Iteration. Daher wird an dieser Stelle auf weitere Ausführungen verzichtet und abschließend noch die diskretisierte Iterationsvorschrift des Parareal-Algorithmus angegeben. Diese setzt sich aus der eigentlichen Iterationsvorschrift

$$U_n^k = \mathscr{G}_{\Delta T}(U_{n-1}^k) + \mathscr{F}_{\Delta T}(U_{n-1}^{k-1}) - \mathscr{G}_{\Delta T}(U_{n-1}^{k-1}) \quad (130)$$

und dem Anfangswert

$$U_0^k = u^0 \quad (131)$$

mit $n \in [1;N]$ und $k \in [1;K_{max}]$ zusammen, wobei k die Anzahl der Iterationen des feinen Propagators repräsentiert [61].
Eine wichtige Erkenntnis, die auch aus den Iterationen von Abbildung 28 hervorgeht, ist die Konvergenz der korrigierten Vorhersagen und der Auswertung des feinen Propagators auf diesen gegen die Referenzlösung. Es liegt eine iterative Konvergenz gegen die Lösung vor, die man bei rein sequentieller Ausführung des feinen Propagators über das gesamte Zeitintervall erhält. Dies ist eine bemerkenswerte Eigenschaft, die man sich im heutigen Zeitalter der Parallelprogrammierung zu Nutze macht [61].

A.3. PFASST

Wie schon im Ausblick, im Kapitel 5.2, erwähnt, setzt sich PFASST aus SDC, Parareal, hybridem Parareal/SDC und FAS zusammen. In Abbildung 29, die dem Vortrag 'From SDC to PFASST' von Speck, Ruprecht, Emmett entnommen ist [74], ist ein dreistufiger V-Zyklus von MLSDC zu sehen.

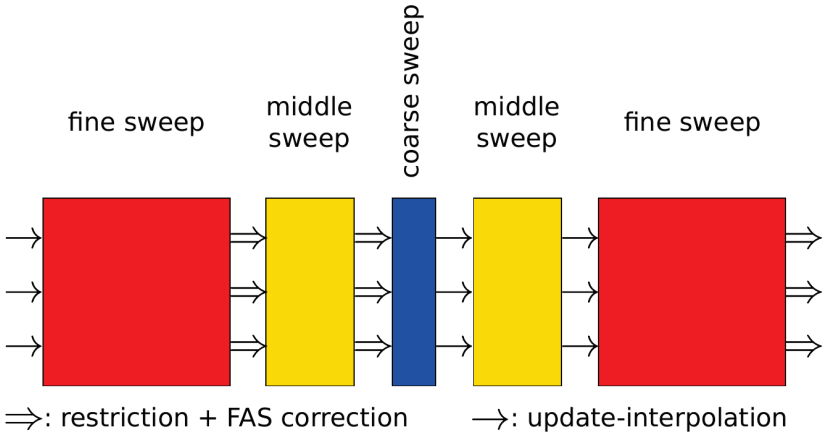


Abbildung 29: Visualisierung der Stufen von MLSDC

Nach dem Initialisierungsvorgang wird zunächst ein feiner Sweep durchgeführt. Der feine Propagator ist hier wie der aus Parareal A.2 definiert und arbeitet mit dem SDC-Verfahren. Das bedeutet, dass nach einem feinen SDC-Sweep vorläufige, neue Werte vorliegen. Diese feinen Werte müssen nun auf gröbere Knoten restringiert werden. In dem Zusammenhang erfolgt eine FAS-Korrektur. Dann wird ein weiterer SDC-Sweep durchgeführt, der nicht so fein ist wie die Auflösung des feinen Propagators, aber auch nicht so grob ist wie die des groben Propagators. Es gibt hier eine Hierarchie von Stufen, auf denen SDC das Problem löst.
Danach erfolgt, wie nach dem feinen SDC-Sweep, erneut eine Restriktion auf noch gröbere Knoten mit zusätzlicher FAS-Korrektur. Zur Ausführung des groben Propagators[66] werden zusätzlich neue Anfangswerte vom voranstehenden Prozessor P_{n-1} genutzt. Dies ist sehr schön in Abbildung 30 zu sehen. Auf die Verknüpfung der einzelnen Prozessoren wird im nächsten Abschnitt dieses Kapitels genauer eingegangen.
Nachdem also der grobe SDC-Sweep ausgeführt wurde, wird die Lösung interpoliert und bisherige Lösungen damit verbessert. Dieser Interpolations- und Korrekturvorgang wird im Übergang vom mittleren Sweep zum feinen Sweep noch einmal wiederholt. Damit ist eine Einheit im MLSDC-Algorithmus abgeschlossen und der nächste Lauf beginnt wieder mit einem feinen SDC-Sweep. Es findet jeweils eine Restriktion und/oder Interpolation in Raum und Zeit statt. Details zum MLSDC-Algorithmus und zum PFASST-Algorithmus sind in [57] nachzulesen.
Zudem ist zu erwähnen, dass die Breite der farbigen Blöcke, die die einzelnen SDC-Sweeps visualisieren, in etwa für den Aufwand stehen soll, der durch sie erzeugt wird [75].

[66] Hier der mittlere, dünne, blaue Block.

Die Parallelität von PFASST wird erst in Abbildung 30, welche ebenfalls aus dem Vortrag von Speck et al. [74] stammt, offensichtlich.
In der Abbildung ist die parallele Arbeit von vier Prozessoren dargestellt. Jeder Prozessor benötigt zur Ausführung von SDC-Sweeps einen Anfangswert zu seiner Startzeit $t_0, t_1, t_2, t_3, \ldots$ Diese Anfangsbedingungen werden durch einen groben und billigen Propagator weitergegeben. Das bedeutet, dass zunächst eine Initialisierungsphase stattfindet, die in der Zeit und damit im Aufwand proportional zur Anzahl der verwendeten Prozessoren wächst. Der Ablauf dieser Initialisierungsphase wird detailliert in [57] beschrieben.
Danach sind in Abbildung 30 die Abfolgen der SDC-Sweeps unterschiedlicher Auflösung aus Abbildung 29 wiederzufinden. Doch auch nach der Initialisierung, die Kommunikation zwischen den einzelnen Prozessoren erfordert, arbeiten die Prozessoren bei der Lösung des Anfangswertproblems nicht völlig autark. Das Ergebnis des feinen SDC-Sweeps von Prozessor P_{n-1} geht in die Ausführung der nächsten Iteration von Prozessor P_n mit ein. Ebenso gilt dies für den mittleren und groben SDC-Sweep. Die Ergebnisse des voranstehenden Prozessors werden auf gleicher Auflösungsstufe an die entsprechenden SDC-Sweeps des nachstehenden Prozessors kommuniziert. Hier kann man von einem Sende-Vorgang sprechen. Dies ist jedoch nur bei der Restriktion und FAS-Korrektur der Fall. Bei der Rückrichtung erfolgt ein Empfangs-Vorgang, der durch eine Interpolation auf feinere Gitter geprägt ist. Auf der gröbsten Stufe kann eine direkte Kommunikation erfolgen, das heißt, dass gleichzeitig gesendet und empfangen werden kann [57, 59, 74].

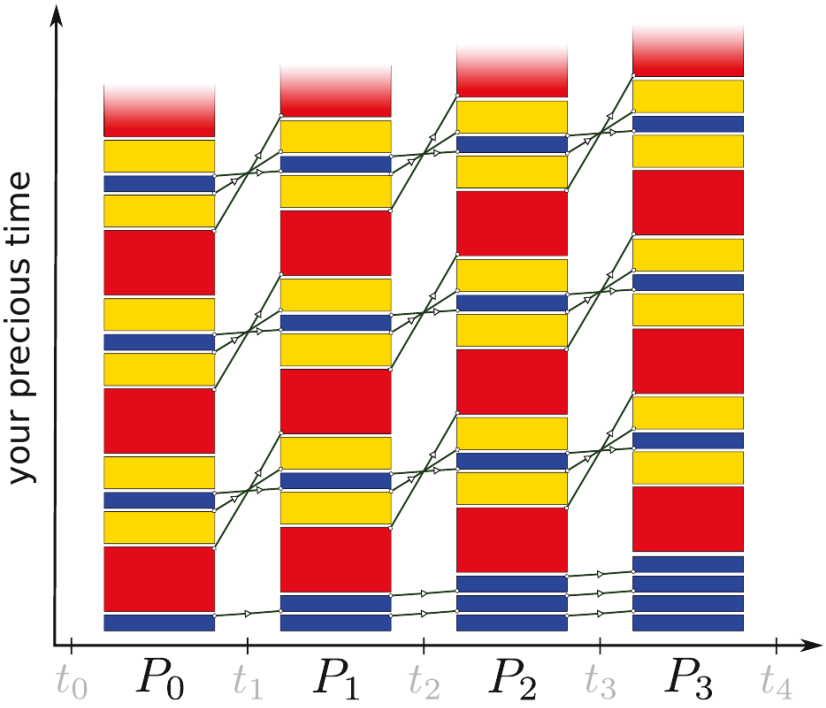


Abbildung 30: Visualisierung der Parallelisierung bei PFASST

Die genaue Herleitung des PFASST-Algorithmus und zusätzliche Informationen zu FAS etc. sind [57] zu entnehmen.

Literatur

[1] Robert G. Williams C. Reid Nichols. *Encyclopedia Of Marine Science*. Facts On File, Inc., 2009.

[2] Wolfgang Bürger. *Der Traum des Seglers bei Flaute*. Birkhäuser, 1998.

[3] Robert Lauterbach. *Physik des Planeten Erde: Ergebnisse geophysikalische Forschung*. Akademie-Verlag, 1975.

[4] Kurt Stüwe. *Einführung in die Geodynamik der Lithosphäre*. Springer, 2000.

[5] Zhongqing Su; Lin Ye. *Identification of Damage Using Lamb Waves*. 48. Springer London, 2009.

[6] Almut Gassmann; Hans-Joachim Herzog. A Consistent Time-Split Numerical Scheme Applied to the Nonhydrostatic Compressible Equations. *Monthly Weather Review*, 135:20–36, 2007.

[7] Andrei Bourchtein; Ludmila Bourchtein. Semi-Lagrangian semi-implicit time-splitting scheme for the shallow water equations. *International Journal for Numerical Methods in Fluids*, 54:453–471, 2007.

[8] Jeffrey S. Whitaker; Sajal K. Kar. Implicit-Explicit Runge-Kutta Methods for Fast-Slow Wave Problems. *Monthly Weather Review*, 141:3426–3434, 2013.

[9] Louis J. Wicker. A Two-Step Adams-Bashforth-Moulton Split-Explicit Integrator for Compressible Atmospheric Models. *Monthly Weather Review*, 137:3588–3595, 2009.

[10] Stefan Jebens; Oswald Knoth; Rüdiger Weiner. Explicit Two-Step Peer Methods for the Compressible Euler Equations. *Monthly Weather Review*, 137:2380–2392, 2009.

[11] Michael Baldauf. Linear Stability Analysis of Runge-Kutta-Based Partial Time-Splitting Schemes for the Euler Equations. *Monthly Weather Review*, 138:4475–4496, 2010.

[12] Sajal K. Kar. A semi-implicit Runge-Kutta time-difference scheme for the two-dimensional shallow-water equations. *Monthly Weather Review*, 134:2916–2926, 2006.

[13] Almut Gassmann. An Improved Two-Time-Level Split-Explicit Integration Scheme for Non-Hydrostatic Compressible Models. *Meteorology and Atmospheric Physics*, 88:23–38, 2005.

[14] Louis J. Wicker; William C. Skamarock. Time-Splitting Methods for Elastic Models Using Forward Time Schemes. *Monthly Weather Review*, 130:2088–2097, 2002.

[15] Dale R. Durran; Peter N. Blossey. Implicit-Explicit Multistep Methods for Fast-Wave-Slow-Wave Problems. *Monthly Weather Review*, 140:1307–1325, April 2012.

[16] Jörg Wensch; Oswald Knoth; Alexander Galant. Multirate Infinitesimal Step Methods for Atmospheric Flow Simulation. *BIT Numerical Mathematics*, 49:449–473, 2009.

[17] Hans Joachim Oberle. Numerik gewöhnlicher Differentialgleichungen. Skript Uni-Hamburg, WS 2008/09.

[18] Josef Stoer; Roland Bulirsch. *Numerische Mathematik 2*. Springer, 2005.

[19] Martin Reißel. Vorlesungsskript der FH-Aachen: Numerik für Differentialgleichungen I, September 2012.

[20] Dale R. Durran. *Numerical Methods for Wave Equations in Geophysical Fluid Dynamics*. Springer, 1998.

[21] Yoshio Kurihara. On the Use of Implicit and Iterative Methods for the Time Integration of the Wave Equation. *Monthly Weather Review*, 23(93):33–46, 1965.

[22] Robert Plato. *Numerische Mathematik kompakt: Grundlagenwissen für Studium und Praxis*, volume 4. Vieweg+Teubner Verlag / GWV Fachverlage GmbH, Wiesbaden, 2010.

[23] Hans Rudolf Schwarz; Norbert Köckler. *Numerische Mathematik*. Vieweg+Teubner, 7th edition, 2009.

[24] Peter Deuflhard; Folkmar A. Bornemann. *Numerische Mathematik II, Gewoehnliche Differentialgleichungen*. de Gruyter Lehrbuch, 2002.

[25] Scott R. Fulton. Semi-Implicit Time Differencing. Technical Report No. 2002-01 Department of Mathematics and Computer Science Clarkson University, Potsdam, NY 13699-5815, Juni 2004.

[26] Michael L. Minion. Semi-Implicit Spectral Deferred Correction Methods for Ordinary Differential Equations. *Communications in Applied Mathematics and Computational Science*, 1(3):471–500, 2003.

[27] Joseph B. Klemp; Robert. B. Wilhelmson. The Simulation of Three-Dimensional Convective Storm Dynamics. *Journal of the Atmospheric Sciences*, 35:1070–1096, 1978.

[28] William C. Skamarock; Joseph B. Klemp. The Stability of Time-Split Numerical Methods for the Hydrostatic and the Nonhydrostatic Elastic Equations. *Monthly Weather Review*, 120:2109–2127, 1992.

[29] Louis J. Wicker; William C. Skamarock. A Time-Splitting Scheme for the Elastic Equations Incorporating Second-Order Runge-Kutta Time Differencing. *Monthly Weather Review*, 126:1992–1999, 1998.

[30] R. James Purser. Accuracy Considerations of Time-Splitting Methods for Models Using Two-Time-Level Schemes. *Monthly Weather Review*, 135:1158–1164, 2007.

[31] Alok Dutt; Leslie Greengard; Vladimir Rokhlin. Spectral Deferred Correction Methods for Ordinary Differential Equations. *BIT Numerical Mathematics*, 40(2):241–266, 2000.

[32] Jun Jia; Judith C. Hill; Katherine J. Evans; George I. Fann; Mark A. Taylor. A Spectral Deferred Correction Method Applied to the Shallow Water Equations on a Sphere. *Monthly Weather Review*, 141:3435–3449, 2013.

[33] Catalin Trenchea. Unconditional Stability of a Partitioned IMEX Method for Magnetohydrodynamic Flows. *Applied Mathematics Letters*, 27:97–100, 2014.

[34] Michael L. Minion. Semi-Implicit Projection Methods for Incompressible Flow Based on Spectral Deferred Corrections. *Applied Numerical Mathematics*, 48:369–387, 2004.

[35] Leandro D. Gryngarten; Andrew Smith; Suresh Menon. Discontinuous Galerkin method with the spectral deferred correction time-integration scheme and a modified moment limiter for adaptive grids. *Communications in Applied Mathematics and Computational Science*, 7:133–174, 2013.

[36] Thomas Y. Hou; Jingfang Huang; Christopher E. Kees; Carl T. Kelley; Hans Petter Langtangen Cass T. Miller; Clint N. Dawson; Matthew W. Farthing. Numerical simulation of water resources problems: Models, methods, and trends. *Advances in Water Resources*, 51:405–437, 2013.

[37] Gregory Forest; Qi Wang; Ruhai Zhou. Kinetic Theory and Simulations of Active Polar Liquid Crystalline Polymers. *Soft Matter*, 9:5207–5222, 2013.

[38] Jia Xin; Jianfei Huang; Weijia Zhao; Jiang Zhu. A Spectral Deferred Correction Method for Fractional Differential Equations. *Abstract and Applied Analysis*, 2013(139530), 2013.

[39] Divya Garg; Micheal A. Patterson; William W. Hager; Anil V. Rao; David Benson; Geoffrey T. Huntington. An Overview Of Three Pseudospectral Methods For The Numerical Solution Of Optimal Control Problems. Preprint AAS 09-332, 2010.

[40] Kenneth L. Judd. *Numerical Methods in Economics*. Massachusetts Institute of Technology Press, 1998.

[41] Aaditya Rangan. Deferred Correction Methods for Low Index Differential Algebraic Equations. In Kluwer Academic Publishers, editor, *BIT Numerical Mathematics*, volume 43, 2003.

[42] Anita T. Layton; Michael L. Minion. Implications of the Choice of Quadrature Nodes for Picard Integral Deferred Corrections Methods for Ordinary Differential Equations. *BIT Numerical Mathematics*, 45:341–373, 2005.

[43] Anders Christian Hansen; John Strain. Convergence Theory for Spectral Deferred Correction. *Numerische Mathematik Manuskript*, 2005.

[44] Andreas Glaser; Vladimir Rokhlin. A New Class of Highly Accurate Solvers for Ordinary Differential Equations. *Journal of Scientific Computing*, 38:368–399, 2009.

[45] Jingfang Huang; Jun Jia; Micheal Minion. Accelerating the Convergence of Spectral Deferred Correction Methods. *Journal of Computational Physics*, 214:633–656, 2006.

[46] Christian Teske. Integralgleichungen und ihre Klassifikation. *Vorlesungsskript der Universität Frankfurt*, WS 2011/2012.

[47] Abdul-Majid Wazwaz. *A First Course in Integral Equations*. World Scientific Publishing Co. Pte. Ltd., 1997.

[48] Thomas Hagstrom; Ruhai Zhou. On the Spectral Deferred Correction of Splitting Methods for Initial Value Problems. *Communications in Applied Mathematics and Computational Science*, 1:169–205, 2006.

[49] Alfio Quarteroni; Riccardo Sacco; Fausto Saleri. *Numerische Mathematik. 2 (2002)*. Springer, 2002.

[50] Robert Schaback; Holger Wendland. *Numerische Mathematik*. Springer, 2005.

[51] Paul Muljadi. *Interpolation - an Overview*. e-book, 2012.

[52] Josef Stoer. *Numerische Mathematik 1*. Springer, 8th edition, 2005.

[53] Norbert Herrmann. *Höhere Mathematik für Ingenieure, Physiker und Mathematiker*, volume 2. Oldenburg, 2007.

[54] Claudio Canuto; M. Youssuff Hussaini; Alfio Quarteroni; Thomas A. Zang. *Spectral Methods*. Springer-Verlag, 2006.

[55] Gene H. Golub; John H. Welsch. Calculation of Gauss Quadrature Rules. *Mathematics of Computation*, 23(106):221–230, 1969.

[56] Robert Speck; Daniel Ruprecht; Matthew Emmett; Michael L. Minion; Matthias Bolten; Rolf Krause. A Multi-Level Spectral Deferred Correction Method. 2013.

[57] Matthew Emmett; Michael L. Minion. Toward an Efficient Parallel in Time Method for Partial Differential Equations. *Communications in Applied Mathematics and Computational Science*, 7:105–132, 2012.

[58] Achi Brandt. Multi-Level Adaptive Solutions to Boundary-Value Problems. *Mathematics of Computation*, 31:333–390, 1977.

[59] Robert Speck; Daniel Ruprecht; Rolf Krause; Matthew Emmett; Michael L. Minion; Matthias Winkel; Paul Gibbon. Integrating an N-Body Problem with SDC and PFASST. *arXiv:1307.1312*, (2012-06), Oktober 2012.

[60] Jacques Louis Lions; Yvon Maday.; Gabriel Turinici. A "Parareal" in Time Discretization of PDE's. *Comptes Rendus de l'Academie des Sciences Series I Mathematics*, 332:661–668, 2001.

[61] Allan S. Nielsen. Feasibility Study of the Parareal Algorithm. Master's thesis, Technical University Denmark, 2012.

[62] Yvon Maday; Gabriel Turinici. The Parareal in Time Iterative Solver: a Further Direction to Parallel Implementation. *Lecture Notes in Computer Science*, 1:441–448, 2003.

[63] Jürg Nievergelt. Parallel Methods for integrating Ordinary Differential Equations. *Communications of the Association for Computing Machinery*, 7(12):731–733, 1964.

[64] Martin J. Gander; Ernst Hairer. *Nonlinear Convergence Analysis for the Parareal Algorithm*, volume 60 of *Lecture Notes in Computational Science and Engineering*. Springer Berlin Heidelberg, 2008.

[65] Martin J. Gander; Stefan Vandewalle. Analysis of the Parareal Time-Parallel Time-Integration Method. *SIAM Journal on Scientific Computing*, 29:556–578, 2007.

[66] Gunnar Andreas Staff; Einar M. Ronquist. *Stability of the Parareal Algorithm*, volume 40 of *Lecture Notes in Computational Science and Engineering*. Springer Berlin Heidelberg, 2005.

[67] Guillaume Bal; Yvon Maday. *A "Parareal"Time Discretization for Non-Linear PDE's with Application to the Pricing of an American Put*, volume 23 of *Lecture Notes in Computational Science and Engineering*. Springer Berlin Heidelberg, 2002.

[68] Prasenjit Saha; Joachim Stadel; Scott Tremaine. A Parallel Integration Method for Solar System Dynamics. *Astrophysics, Cornell University*, 1996.

[69] Leonardo Baffico; Samuel Bernard; Yvon Maday; Gabriel Turinici; Gilles Zérah. Parallel-In-Time Molecular-Dynamics Simulations. *Physical Review E: Statistical, Nonlinear, and Soft Matter Physics*, 66:057701–1–057701–4, Nov. 2002.

[70] Michael L. Minion; Sarah A. Williams. Parareal and Spectral Deferred Corrections. *AIP Conference Proceedings*, 1048:388–391, 2008.

[71] Michael L. Minion. A Hybrid Parareal Spectral Deferred Corrections Method. *Communications in Applied Mathematics and Computational Science*, 5:265–301, 2010.

[72] Daniel Ruprecht; Rolf Krause. Explicit Parallel-In-Time Integration of a Linear Acoustic-Advection System. *Computers & Fluids*, 59:72–83, 2011.

[73] Martin J. Gander. Analysis of the Parareal Algorithm Applied to Hyperbolic Problems Using Characteristics. *Boletin de la Sociedad Espanola de Matematica Aplicada*, 42:21–35, 2008.

[74] Robert Speck; Daniel Ruprecht; Matthew Emmett. From SDC to PFASST; unpublished. Juni 2013.

[75] Robert Speck; Daniel Ruprecht; Rolf Krause; Matthew Emmett; Michael L. Minion; Matthias Winkel; Paul Gibbon. A Massively Space-Time Parallel N-Body Solver. *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis; SC '12; IEEE Computer Society Press*, (92):92:1–92:11, 2012.

Jül-4377
Juli 2014
ISSN 0944-2952

JÜLICH
FORSCHUNGSZENTRUM